Handbuch der BASIC-Dialekte

Lexikon

INHALT

ABS
ACOS
AGS
ADR
AFTER GOSUB
ALLOCATE
AND
ANGLE
APPEND
ASC
ASIN
ASK
ASN
ATN
ATTR
AUTO
BACKUP
BANK
BASE
BEEP
BEGIN BEND
BIN
BIN$
BLOAD
BOOT
BORDER
BOX
BRIGHT
BSAVE
BUMP
BYE
CALL
CAPTURE
CASE
CAT
CATALOG
CAUSE EXCEPTION
CDBL
CEIL
CHAIN
CHAIN MERGE
CHAR
CHR$
CINT
CIRCLE
CIRCLE
CLEAR
CLG
CLOAD
CLOAD
CLOG
CLOSE
CLOSEIN
CLOSEOUT
CLR
CLS
CMD
CODE
CODE
COLLECT
COLLISION
COLOR
COLOR
COLOR
COM
CONT
CONTINUE
COPY
COPY
COS
COSH
COT
CREAL
CSAVE
CSAVE
CSC
CSNG
CSRLIN
DATA
DATE
DATE$
DCLEAR
DEC
DEF
DEFDBL
DEF FN
DEFINT
DEFREAL
DEFSNG
DEFSTR
DEFUSR
DEG
DEG
DELETE
DI EI
DIM
DIRECTORY
DISPLAY
DLOAD
DO LOOP
DOS
DRAW
DRAWR
DRAWTO
DS
DS$
DSAVE
EDIT
EI
EL
ELSE
END
ENT
ENTER
ENV
ENVELOPE
EOF
EPS
ER
ERASE
ERL
ERR
ERR$
ERROR
EVERY GOSUB
EXIT
EXLINE
EXP
EXSTRING$
EXT
EXTYPE
FAST
FETCH
FILTER
FIX
FKEY
FLASH
FLUSH
FN
FOR TO STEP
FORMAT
FP
FRE
FREE
GET
GET#
GETKEY
GO64
GOSUB RETURN
GOTO
GRAPHIC
GRAPHICS
GRAPHICS
GSHAPE
HANDLER
HEADER
HELP
HEX $
HIMEM
IF THEN ELSE
IMAGE
IN
INF
INFO
INK
INKEY
INKEY$
INP
INPUT
INPUTS
INPUT#
INSTR
INT
INTERVALL ON/OFF/STOP
INVERSE
IP
JOY
KEY
KEY DEF
KEY LIST
KEY ON/OFF
KEY (X) ON/OFF/STOP
LBOUND
LCASE$
LEFT$
LEN
LET
LINE
LINE INPUT
LIST
LLIST
LN
LOAD
LOCATE
LOG
LOG2
LOG10
LOOK
LOOP
LOWER$
LPOS
LPRINT
LTRIM$
MAX
MAXLEN
MEMORY
MERGE
MID$
MID$ MIN
MOD
MODE
MONITOR
MOTOR
MOVE
MOVER
MOVSPR
NEW
NEW ALL
NEXT
NOT
NOTE
NUMERIC
OCT$
ON BREAK GOSUB
ON BREAK STOP
ON ERROR GOTO
N GOSUB
ON GOTO

Fortsetzung letzte Seite

S

Handbuch der BASIC-Dialekte

Peter Fritzsche
Georg Seeßlen

(1986)

Lexikon

CIP-Kurztitelaufnahme der Deutschen Bibliothek

Fritzsche, Georg:
Handbuch der BASIC-Dialekte / [Peter Fritzsche; Georg Seeßlen]. –
München: F. Schneider, 1986.
(Schneider-Buch: Lexikon)
ISBN 3-505-09206-1
NE: Seeßlen, Georg; HST

87
6249

Stadt- und
Landesbibliothek
Dortmund

© 1986 by Franz Schneider Verlag GmbH
8000 München 46 · Frankfurter Ring 150
Alle Rechte vorbehalten
Umschlagbild: Prof. Ernst Jünger, München
Umschlaggestaltung: Angelika Bachmann, München
Illustrationen: Otto Wamser, München
Lektorat/Redaktion: Tomas de Lates
Herstellung: Stefan Prochnow
Druck: Presse-Druck Augsburg
ISBN: 3 505 09206-1
Bestell-Nr.: 9206

Vorwort

BASIC ist immer noch in. Es ist zwar langsam und bietet wenig Möglichkeiten zum strukturierten Programmieren, ist jedoch sehr vielseitig und anschaulich. Außerdem ist es leicht zu lernen und führt den Anfänger rasch zu Erfolgserlebnissen. Mittlerweile gibt es allerdings so viele BASIC-Dialekte, und ihr Befehlsvorrat ist zuweilen so umfangreich geworden, daß oft nicht einmal das jeweilige Handbuch sämtliche Befehle enthält. Aufgrund ihres unterschiedlichen Befehlssatzes lassen sich viele Programme nicht von einem auf den anderen Mikrocomputer übertragen - unerfreulich für Umsteiger und alle, die Programme austauschen oder Listings aus Zeitschriften übernehmen wollen.

Das vorliegende Handbuch beschreibt alle BASIC-Befehle der im deutschsprachigen Raum zur Zeit gebräuchlichsten Mikrocomputer und erläutert anhand von Beispielen ihre Anwendungsmöglichkeiten. Zudem erhält der Leser Tips, wie sich bestimmte Befehle auf anderen Rechnern simulieren lassen, in deren BASIC-Dialekt sie eigentlich gar nicht vorgesehen sind. Die Kompatibilitätsliste bei jedem Stichwort zeigt, auf welchen Rechnern der Befehl vorhanden ist und mit welcher Syntax er jeweils verwendet werden muß.

Bei der Auswahl der BASIC-Dialekte bzw. der entsprechenden Mikrocomputer haben wir nicht nur Oldtimer, Dauerbrenner, und erfolgreiche Newcomer berücksichtigt, sondern auch den bislang hierzulande noch wenig populären MSX-Standard sowie schließlich einen Außenseiter, nämlich den Enterprise, der wegen seiner Grafik-Fähigkeiten und außergewöhnlichen Speicherwerweiterungsmöglichkeiten interessant ist.

Mit Hilfe dieses Handbuchs soll die Vielfalt der Programmiermöglichkeiten in BASIC erweitert werden. Der Benutzer soll nicht nur Programme unterschiedlicher Rechner analysieren und umformen sondern seine eigenen Entwürfe gleich für mehrere Computertypen entwickeln können.

Dieses Werk ist für den Einsteiger obendrein ein ergänzendes Lehrbuch. Die kurzen, einleuchtenden Beispiele vermitteln zusätzliche Übungspraxis. Und der Fortgeschrittene findet neue Anregungen.

Wir danken folgenden Firmen, die uns bei der Erstellung dieses Buches unterstützt haben:

Atari

Centronics

Commodore

Enterprise

Philips

Sanyo

Schneider Computer Division

Wenger Datentechnik

Frau Gabriele Ehrich

Helge Puhlmann und Walter Pede

ABS

Befehlsart: Funktion

ABS(**Zahl**)

Beschreibung: Die Funktion wandelt eine **Zahl** in ihren absoluten Wert um. Der absolute Wert einer Zahl gibt ihre Größe ohne das Vorzeichen wieder. Der absolute Wert von -2 ist derselbe wie der von +2, also 2. In der Klammer nach der ABS-Funktion kann sowohl eine Zahl als auch eine vorher definierte Variable, aber auch ein Rechenausdruck stehen.

Beispiele:

```
PRINT ABS(-2.98)
 2.98
READY.
PRINT ABS(10-11.987)
 1.987
READY.
```

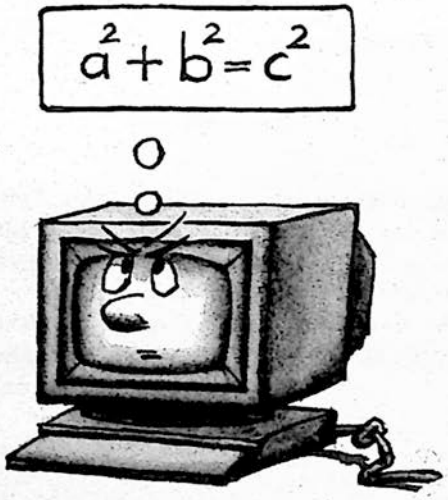
```
A= 45-78
READY.
PRINT ABS(A)
 33
READY.
```

```
10 REM * FORMEL1 *
20 PRINT "WIEVIEL KILOMETER IST WAGEN 1 GEFAHREN"
30 INPUT A
40 PRINT "WIEVIEL KILOMETER IST WAGEN 2 GEFAHREN"
50 INPUT B
60 PRINT "ABSTAND ZWISCHEN DEN BEIDEN WAGEN:
                                    ";ABS(A-B);" KILOMETER."
READY.
```

Anmerkung zur Kompatibilität: Die Funktion ABS gibt es in nahezu jedem BASIC-Dialekt. Beim ZX81 und Spectrum können die Klammern weggelassen werden.

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	ABS(Zahl)
› COMMODORE VC20,64	:	ABS(Zahl)
› COMMODORE 16,116,PLUS4	:	ABS(Zahl)
› COMMODORE 128	:	ABS(Zahl)
› ENTERPRISE 64,128	:	ABS(Zahl)
› MSX-STANDARD	:	ABS(Zahl)
› SCHNEIDER CPC464	:	ABS(Zahl)
› SCHNEIDER CPC664,CPC6128	:	ABS(Zahl)
› SINCLAIR ZX81	:	ABS Zahl
› SINCLAIR SPECTRUM	:	ABS Zahl



ACOS

Befehlsart: Funktion

ACOS(**Zahl**)

Beschreibung: Ergebnis ist die Errechnung der ARCUSCOSINUS-Funktion. Man erhält für die **Zahl**en -1 bis 1 die entsprechenden Winkel der COSINUS-Funktion (siehe COS). Der Winkel hat eine Größe von PI (siehe PI) bis Ø, bzw. von 18Ø bis Ø Grad.

Bei Eingabe einer **Zahl**, die größer als 1 bzw. kleiner als -1 ist, erfolgt eine Fehlermeldung.

Beispiel:

```
PRINT ACOS(Ø.5)
 1.Ø47197551
READY
```

Anmerkung: Um das Ergebnis in Grad zu erhalten, muß man die Ausgabe von PRINT ACOS(**Zahl**) mit 57.29578 multiplizieren.

Beispiel:

```
1Ø PRINT "EINGABE COSINUS" (-1 BIS 1)"
2Ø INPUT C
3Ø A=ACOS (C)
4Ø G= INT(A * 57.29578)
5Ø PRINT "ERGEBNIS IN GRAD: ";G
```

Die Funktion sollte nur verwendet werden, wenn man mit den Winkelfunktionen vertraut ist.

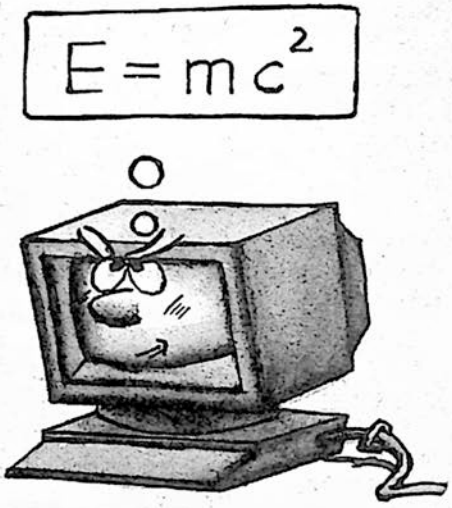
Anmerkung zur Kompatibilität: Die Funktion ACS beim ZX81 und ZX Spectrum entspricht der ACOS-Funktion. Sie hat das Format ACS**Zahl**; das heißt, die Zahl steht ohne Klammern.

Bei Rechnern, die diesen Befehl nicht besitzen, läßt sich das Ergebnis mit folgender Formel erzielen:

-ATN(**Zahl** /SQR(1-**Zahl**↑ 2))+PI/2

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	ACOS(Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	ACS Zahl
SINCLAIR SPECTRUM	:	ACS Zahl



ACS

Befehlsart: Funktion

ACS **Zahl**

Beschreibung: Ergebnis ist die Errechnung der ARCUSCOSINUS-Funktion. Man erhält für die **Zahl**en -1 bis 1 die entsprechenden Winkel der COSINUS-Funktion (siehe COS). Der Winkel hat eine Größe von PI (siehe PI) bis Ø, bzw. von 18Ø bis Ø Grad.

Bei Eingabe einer **Zahl**, die größer als 1 bzw. kleiner als -1 ist, erfolgt eine Fehlermeldung.

Beispiel:

```
PRINT ACS Ø.5
 1.Ø47197551
READY
```

Anmerkung: Um das Ergebnis in Grad zu erhalten, muß man die Ausgabe von PRINT ACS **Zahl** mit 57.29578 multiplizieren.

Beispiel:

```
1Ø PRINT "EINGABE COSINUS (-1 bis 1)"
2Ø INPUT C
3Ø A=ACS C
4Ø G=INT(A * 57.29578)
5Ø PRINT "ERGEBNIS IN GRAD: ";G
```

Die Funktion sollte nur verwendet werden, wenn man mit den Winkelfunktionen vertraut ist.

Anmerkung zur Kompatibilität: Die Funktion ACS entspricht der Funktion ACOS beim "Enterprise". Sie hat dort das Format ACOS(**Zahl**).

Bei Rechnern, die diesen Befehl nicht besitzen, läßt sich das Ergebnis mit folgender Formel erzielen:

-ATN(**Zahl**/SQR(1-**Zahl**↑ 2))+PI/2

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	ACOS(Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
› SINCLAIR ZX81	:	ACS Zahl
› SINCLAIR SPECTRUM	:	ACS Zahl

ADR

Befehlsart: Funktion

ADR(**Variablenname**)

Beschreibung: Mit diesem Befehl erhält man die Speicheradresse, unter der eine Variable abgelegt wurde.
Der **Variablenname** gibt dabei den Namen der Variablen an, dessen Adresse ermittelt werden soll.
Es kann ein kleines Maschinenprogramm in einer Stringvariablen abgelegt werden. Mit der ADR-Funktion läßt sich dessen Anfangsadresse ermitteln.

Beispiel:

```
Y=5Ø
PRINT ADR(Y)
```

Anmerkung: Bei Mehr-Byte-Variablen, also Variablen, die länger als ein BYTE sind, wird die Adresse der ersten BYTES angegeben. Es sind in der Regel Mehr-Byte-Variablen zu bearbeiten. Ausnahme ist z. B. die INTEGER-Variable, also die Ganzzahl-Variable.

Kompatibilitätsliste:

› ATARI 600,800,130XE	: ADR(Variablenname)
COMMODORE VC20,64	:
COMMODORE 16,116,PLUS4	:
COMMODORE 128	:
ENTERPRISE 64,128	:
MSX-STANDARD	:
SCHNEIDER CPC464	:
SCHNEIDER CPC664,CPC6128	:
SINCLAIR ZX81	:
SINCLAIR SPECTRUM	:

AFTER GOSUB

Befehlsart: Anweisung

AFTER **Zahl 1**, **Zahl 2** GOSUB **Zeilennummer**

Beschreibung: Durch diesen Befehl wird nach **Zahl 1** * Ø.Ø2 Sekunden in ein Unterprogramm gesprungen. Bis die Zeit (**Zahl 1** * Ø.Ø2 Sekunden) verstrichen ist, wird das Unterprogramm ausgeführt, das mit RETURN (siehe dort) abgeschlossen wird und ins Hauptprogramm zurückführt.

Die **Zahl 2** spricht eine der vier Uhren im Schneider-Computer an. Sie kann deshalb nur die Werte Ø, 1, 2 oder 3 annehmen. Läßt man die **Zahl 2** und das Komma im Format der Anweisung weg, wird automatisch die Uhr Ø angewählt.

Die **Zeilennummer** gibt die Startzeile (auch "Anfangsadresse" genannt) des Unterprogramms an.

Beispiel:

```
10 AFTER 25 GOSUB 100
20 PRINT "Buecher lesen macht Spass!"
30 GOTO 20
100 PRINT "Schneider-";
110 RETURN
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	AFTER Zahl1,Zahl2 GOSUB Zeilennummer
› SCHNEIDER CPC664,CPC6128	:	AFTER Zahl1,Zahl2 GOSUB Zeilennummer
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

ALLOCATE

Befehlsart: Anweisung

ALLOCATE **Zahl**

Beschreibung: Es wird in einem BASIC-Programm Platz für ein Maschinenprogramm geschaffen. Dabei gibt die **Zahl** in der Anzahl der BYTES die benötigte Größe des freien Platzes für das Maschinenprogramm an.

Beispiel:

```
ALLOCATE 2Ø
```

Anmerkung: Der Befehl ALLOCATE ist nur sinnvoll in Verbindung mit Maschinencode-Unterprogrammen. Er löscht alle Variablen.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	ALLOCATE Zahl
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

AND

Befehlsart: Operator

Argument 1 AND **Argument 2**

Beschreibung: Mit AND lassen sich mehrere Bedingungen miteinander logisch verknüpfen. Das heißt: Bei einer bedingten Verzweigung mit IF...THEN verlangt der logische Operator AND nach der Erfüllung zweier Bedingungen.

Beispiel:

siehe Seite 20

AND vergleicht die beiden Argumente, um zu einem Ergebnis zu kommen:

Argument 1	AND	Argument 2	Ergebnis
wahr (Bedingung erfüllt)		wahr (Bedingung erfüllt)	wahr (Bedingung erfüllt)
wahr (Bedingung erfüllt)		nicht wahr (Bedingung nicht erfüllt)	nicht wahr (Bedingung nicht erfüllt)
nicht wahr (Bedingung nicht erfüllt)		wahr (Bedingung erfüllt)	nicht wahr (Bedingung nicht erfüllt)
nicht wahr (Bedingung nicht erfüllt)		nicht wahr (Bedingung nicht erfüllt)	nicht wahr (Bedingung nicht erfüllt)

AND

Die Argumente bei einem Mehrfachvergleich mit AND können nicht nur aus Zahlen-Operationen, sondern auch aus Textausdrücken bestehen.

Beispiel:

siehe Seite 20

Intern bearbeitet der Computer eine logische Operation mit AND, indem er die beiden Argumente (Operatoren) in ganze 16-Bit-Binärzahlen umwandelt. Der Vergleich wird nun Bit für Bit durchgeführt.

```
PRINT 15 AND 14
```

ergibt:

```
 14
```

denn

```
          15 = ØØØØØØØØØØØØ1111
AND
          14 = ØØØØØØØØØØØØ111Ø

ergibt         ØØØØØØØØØØØØ111Ø
```

In eine logische Operation übersetzt bedeutet dieser Vergleich:

	AND		ergibt	
nicht wahr	(Ø)	nicht wahr (Ø)	nicht wahr	(Ø)
nicht wahr	(Ø)	nicht wahr (Ø)	nicht wahr	(Ø)
nicht wahr	(Ø)	nicht wahr (Ø)	nicht wahr	(Ø)
nicht wahr	(Ø)	nicht wahr (Ø)	nicht wahr	(Ø)
nicht wahr	(Ø)	nicht wahr (Ø)	nicht wahr	(Ø)
nicht wahr	(Ø)	nicht wahr (Ø)	nicht wahr	(Ø)
nicht wahr	(Ø)	nicht wahr (Ø)	nicht wahr	(Ø)
nicht wahr	(Ø)	nicht wahr (Ø)	nicht wahr	(Ø)
nicht wahr	(Ø)	nicht wahr (Ø)	nicht wahr	(Ø)
nicht wahr	(Ø)	nicht wahr (Ø)	nicht wahr	(Ø)
nicht wahr	(Ø)	nicht wahr (Ø)	nicht wahr	(Ø)
nicht wahr	(Ø)	nicht wahr (Ø)	nicht wahr	(Ø)
wahr	(1)	wahr (1)	wahr	(1)
wahr	(1)	wahr (1)	wahr	(1)
wahr	(1)	wahr (1)	wahr	(1)
wahr	(1)	nicht wahr (Ø)	nicht wahr	(Ø)

Wie die anderen logischen Operatoren (siehe OR, siehe NOT, siehe XOR) dient auch AND zum Testen von Mehrfachvergleichen, aber auch zur Bit-Manipulation und zur Durchführung Boolescher Operationen. AND darf daher keinesfalls mit einem Plus (+) oder einer Anweisung zum Zusammenfügen von Textausdrücken verwechselt werden.

Beispiel:

siehe Seite 20

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       : Argument1 AND Argument2
> COMMODORE VC20,64         : Argument1 AND Argument2
> COMMODORE 16,116,PLUS4    : Argument1 AND Argument2
> COMMODORE 128             : Argument1 AND Argument2
  ENTERPRISE 64,128         :
> MSX-STANDARD              : Argument1 AND Argument2
> SCHNEIDER CPC464          : Argument1 AND Argument2
> SCHNEIDER CPC664,CPC6128  : Argument1 AND Argument2
> SINCLAIR ZX81             : Argument1 AND Argument2
> SINCLAIR SPECTRUM         : Argument1 AND Argument2
```

```
10 INPUT "BITTE EINE ZAHL";A
20 IF A>0 AND A<100 THEN PRINT "DIE ZAHL LIEGT ZWISCHEN 1 UND 100":END
30 PRINT "DIE ZAHL IST GROESSER ALS HUNDERT,"
40 PRINT "ODER ES IST EINE NEGATIVE ZAHL, BZW. 0."
```

```
10 PRINT "␣GIB ZWEI ZAHLEN EIN.":PRINT"DIE ERSTE GROESSER,"
20 PRINT "DIE ZWEITE KLEINER. ABER NICHT"
30 PRINT "KLEINER ALS SECHS."
40 INPUT A
50 INPUT B
60 IF A>B AND B>5 THEN C=A-B:GOTO 80
70 PRINT "FALSCHE EINGABE. NOCHMAL:":GOTO 40
80 PRINT "WIEVIEL IST ";A;" - ";B
90 INPUT D
100 IF D=C THEN PRINT "RICHTIG!":END
110 PRINT "LEIDER FALSCH. NOCHMAL:":GOTO 90
```

```
10 PRINT "FUEHLEN SIE SICH GUT"
20 INPUT G$
30 PRINT "SIND SIE MUEDE"
40 INPUT M$
50 IF G$="JA" AND M$="NEIN" THEN PRINT "␣DANN MAL LOS!":GOTO 70
60 PRINT "SCHADE. DANN AUF EIN ANDERMAL!":END
70 PRINT"BEGINN DES PROGRAMMES"
```

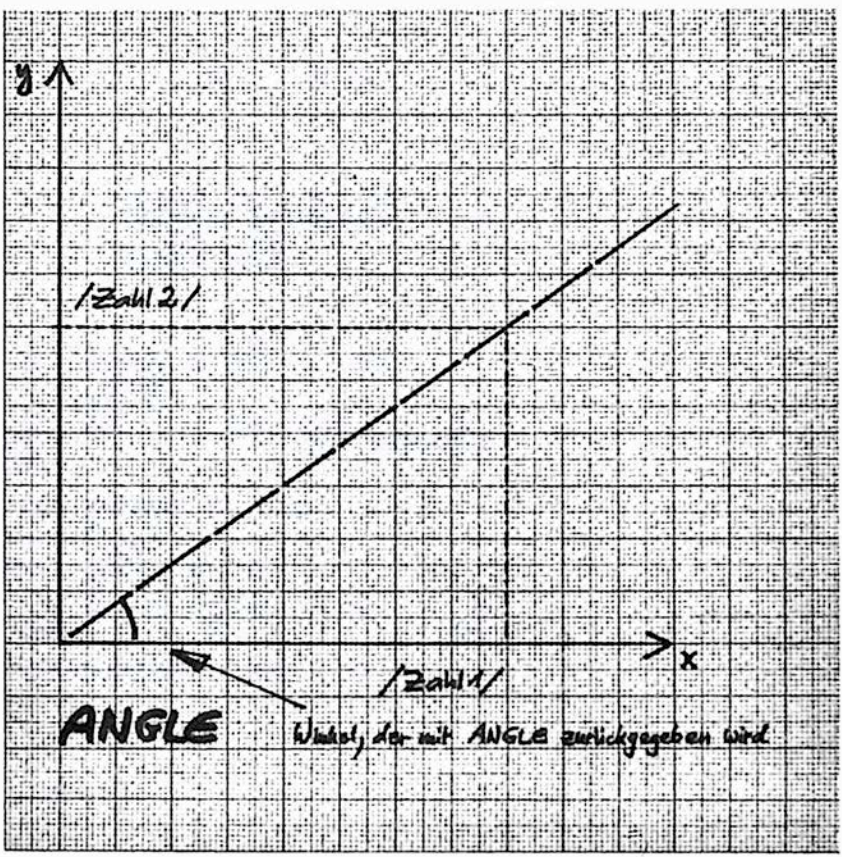
ANGLE

Befehlsart: Funktion

ANGLE(**Zahl 1,Zahl 2**)

Beschreibung: Mit diesem Befehl ermittelt man den Winkel zwischen der waagrechten x-Achse und der Geraden, die durch die Punkte (Ø,Ø) und (**Zahl 1,Zahl 2**) geht.

Die **Zahl 1** gibt dabei die x-Koordinate, die **Zahl 2** die y-Koordinate des zweiten Geradenpunkts an:



ANGLE

Beispiel:

```
10 INPUT "X-KOORDINATE";X
20 INPUT "Y-KOORDINATE";Y
30 PRINT ANGLE(X,Y)
40 GOTO 10
```

Anmerkung: Die Funktion sollte man nur verwenden, wenn man mit Winkelfunktionen vertraut ist.

Anmerkung zur Kompatibilität: Bei Rechnern, die über diesen Befehl nicht verfügen, läßt sich das Ergebnis folgendermaßen erreichen:

ANGLE(**Zahl 1,Zahl 2**)

wird ersetzt durch

ATN (**Zahl 2/Zahl 1**)

Zahl 1 darf bei dieser Umschreibung nicht Ø werden.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	ANGLE(Zahl1,Zahl2)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

APPEND

Befehlsart: Anweisung

APPEND **Zahl 1**,"**Dateiname**",D**Zahl 2**,U**Zahl 3**

Beschreibung: Mit diesem Befehl wird eine unter dem **Dateinamen** bestehende, sequentielle Datei geöffnet und der Datenzeiger an das Ende dieser Datei positioniert. Danach lassen sich neue Daten an die Datei **Dateiname** anfügen.

Auf diese Weise sind größere Dateien möglich, als es normalerweise bei sequentiellen Dateien der Fall ist. Diese können nur so groß sein wie der Rechnerspeicher. Will man nämlich eine sequentielle Datei verlängern, muß man diese erst in den Rechner einlesen, um daraufhin die verlängerte Datei aufs neue abzuspeichern.

Die **Zahl 1** ist ein ganzzahliger Wert im Bereich von Ø bis 255. Dieser Wert wird der so eröffneten Datei als Kennummer zugeordnet. Man nennt diese Zahl auch logische File-Nummer. Bei Nummern, die kleiner als 128 sind, wird automatisch ein Ende der Daten ausgegeben. Bei größeren Nummern muß das Ende von Daten durch CHR$(13) (siehe CHR$) gekennzeichnet werden.

Mit **Zahl 1** wird z. B. beim PRINT-Befehl (siehe PRINT) die Datei mit **Dateiname** angesprochen.

Der **Dateiname** gibt den Namen der Datei wieder, die geöffnet werden soll. Er kann maximal 16 Zeichen lang sein. Statt des Dateinamens kann auch eine Stringvariable verwendet werden, in der der Dateiname belegt ist. Dabei entfallen die Anführungszeichen.

Zahl 2 gibt die Nummer des Laufwerkes an. Sie kann entweder Ø oder 1 betragen. Wird die Zahl weggelassen, so wird automatisch der Wert Ø angenommen.

Zahl 3 gibt die Geräteadresse der angeschlossenen Floppydisk an. Sie ist nur von Bedeutung, wenn man mehrere Ein-zellaufwerke verwendet.

Beispiele:

```
APPEND 138, "test"
APPEND 1, a$, DØ
APPEND 2,"VERSUCH",DØ,U)
```

Kompatibilitätsliste:

```
ATARI 600,800,130XE       :
COMMODORE VC20,64         :
COMMODORE 16,116,PLUS4    :
› COMMODORE 128           :
      APPEND#Zahl1,"Dateiname",DZahl2,UZahl3
ENTERPRISE 64,128         :
MSX-STANDARD              :
SCHNEIDER CPC464          :
SCHNEIDER CPC664,CPC6128  :
SINCLAIR ZX81             :
SINCLAIR SPECTRUM         :
```

ASC

Befehlsart: Funktion

ASC(**Textausdruck**)

Beschreibung: Die Funktion gibt den numerischen Wert des ersten Zeichens eines **Textausdruck**es wieder. Der **Textausdruck** muß dabei mit Anführungszeichen versehen sein.

Ein Computer "weiß" nicht, was Buchstaben sind. Daher sind den Buchstaben Zahlen zugeordnet. Mit diesen Zahlen bestimmt er die Adresse, unter der er das Erscheinungsbild der Buchstaben abgespeichert hat. Eine solche Zahlenzuordnung nennt man einen Code. Der Code, der für die Computer verwendet wird, ist der ASCII-Code.

Beispiel:

```
10 FOR I=1 TO 13
20 A$="SCHNEIDERBUCH"
30 PRINT ASC (RIGHT$(A$,I));
40 NEXT I
```

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  ASC(Textausdruck)
> COMMODORE VC20,64         :  ASC(Textausdruck)
> COMMODORE 16,116,PLUS4    :  ASC(Textausdruck)
> COMMODORE 128             :  ASC(Textausdruck)
  ENTERPRISE 64,128         :  ORD(Textausdruck)
> MSX-STANDARD              :  ASC(Textausdruck)
> SCHNEIDER CPC464          :  ASC(Textausdruck)
> SCHNEIDER CPC664,CPC6128  :  ASC(Textausdruck)
  SINCLAIR ZX81             :  CODE Textausdruck
  SINCLAIR SPECTRUM         :  CODE Textausdruck
```

ASIN

Befehlsart: Funktion

ASIN(**Zahl**)

Beschreibung: ASIN(**Zahl 1**) gibt die ARCUSSINUS-Funktion der **Zahl** wieder. Das heißt, man erhält für die **Zahl**en -1 bis 1 die entsprechenden Winkel der SINUS-Funktion (siehe SIN). Der Winkel hat eine Größe von -PI/2 (siehe PI) bis PI/2, bzw. von -9Ø Grad bis 9Ø Grad.

Bei Eingabe einer größeren **Zahl** als 1 oder einer kleineren **Zahl** als -1 erfolgt eine Fehlermeldung.

Beispiel:

```
PRINT ASIN(Ø.5)
 Ø.523598775
READY
```

Anmerkung: Der Einsatz dieses Befehls ist nur sinnvoll, wenn man mit den Winkelfunktionen vertraut ist.

Anmerkungen zur Kompatibilität: Bei Rechnern, die in ihrem BASIC-Dialekt den Befehl ASIN nicht aufweisen, läßt er sich umschreiben:

ASIN (**Zahl**) entspricht
ATN(**Zahl**/SQR(1-**Zahl**↑ 2))

Beim Sinclair ZX 81 und Sinclair Spectrum hat der Befehl das Format

ASN Zahl

(siehe Kompatibilitätsliste).

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	ASIN(Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	ASN Zahl
SINCLAIR SPECTRUM	:	ASN Zahl

ASK

Befehlsart: Funktion

ASK **Maschinenoption Variablenname**

Beschreibung: Mit dieser Funktion werden bestimmte Optionen abgefragt, wie zum Beispiel die Tastaturwiederholungsrate.

Maschinenoption gibt die Option wieder, die abgefragt werden soll. Folgende Optionen sind möglich:

KEY RATE, BIAS, BORDER, STATUS, VIDEO COLOUR, KEY CLICK, KEY DELAY, TIMER, DEFAULT CHANNEL, FAST SAVE, REM1, REM2, TAPE LEVEL, TAPE SOUND, EDITOR BUFFER, EDITOR KEY, EDITOR

Der **Variablenname** bezeichnet die Variable, der der zurückgegebene Wert zugewiesen wird.

Beispiel:

ASK KEY RATE A

In diesem Beispiel wird die Tastaturwiederholungsrate abgefragt und der Variablen A zugewiesen.

Kompatibilitätsliste:

ATARI 600,800,130XE	:
COMMODORE VC20,64	:
COMMODORE 16,116,PLUS4	:
COMMODORE 128	:
› ENTERPRISE 64,128	: ASK Maschinenoption Variable
MSX-STANDARD	:
SCHNEIDER CPC464	:
SCHNEIDER CPC664,CPC6128	:
SINCLAIR ZX81	:
SINCLAIR SPECTRUM	:

Manche Schachprogramme sind schon überraschend leistungsfähig

ASN

Befehlsart: Funktion

ASN **Zahl**

Beschreibung: ASN **Zahl** gibt die ARCUSSINUS-Funktion der **Zahl** wieder. Diese Funktion entspricht der Funktion ASIN(**Zahl**) (siehe ASIN)
(siehe Kompatibilitätsliste).

Beispiel:

```
PRINT ASN Ø.5
 Ø.523598775
READY
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	ASIN(Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
› SINCLAIR ZX81	:	ASN Zahl
› SINCLAIR SPECTRUM	:	ASN Zahl

ATN

Befehlsart: Funktion

ATN(**Zahl**)

Beschreibung: Mit der mathematischen Funktion ARCUSTANGENS erhält man den durch die **Zahl** festgelegten Winkel im Bogenmaß. Der Wert kann zwischen -PI/2 und PI/2 liegen (siehe PI).

Statt einer **Zahl** läßt sich auch ein Rechenausdruck oder eine vorher definierte Variable verwenden.

Beispiele:

```
WERT
PRINT ATN (2)
 1.10714872
READY
```

```
10 PRINT ATN (1)
20 FOR a=1 TO 8
30 PRINT ATN (a)
40 NEXT a
10 PRINT ATN (1),
20 FOR a=1 TO 8
30 PRINT ATN (a)
40 NEXT a
50 B=2
60 PRINT:PRINT ATN (A-B)
```

Anmerkung: Die Verwendung der Funktion ist nur sinnvoll, wenn man mit den Winkelfunktionen vertraut ist.

Anmerkung zur Kompatibilität: Mit der ARCUSTANGENS-Funktion ATN lassen sich eine Reihe anderer Funktionen umschreiben (siehe ACS, siehe ASN).

Kompatibilitätsliste:

```
› ATARI 600,800,130XE        :  ATN(Zahl)
› COMMODORE VC20,64          :  ATN(Zahl)
› COMMODORE 16,116,PLUS4     :  ATN(Zahl)
› COMMODORE 128              :  ATN(Zahl)
› ENTERPRISE 64,128          :  ATN(Zahl)
› MSX-STANDARD               :  ATN(Zahl)
› SCHNEIDER CPC464           :  ATN(Zahl)
› SCHNEIDER CPC664,CPC6128   :  ATN(Zahl)
› SINCLAIR ZX81              :  ATN Zahl
› SINCLAIR SPECTRUM          :  ATN Zahl
```

ATTR

Befehlsart: Funktion

ATTR(**Zahl 1**,**Zahl 2**)

Beschreibung: Mit dieser Funktion wird eine bestimmte, durch **Zahl 1** und **Zahl 2** bezeichnete Stelle auf dem Bildschirm nach Helligkeit, Farbe, Blinken/Nichtblinken abgefragt.

Der Bildschirm ist in zwei Speicherbereichen im Rechner abgelegt. Der eine Speicherbereich enthält die Zeichen, die auf dem Bildschirm dargestellt werden. Der zweite Bereich enthält die Informationen über die Hintergrundfarbe, Schriftfarbe, Helligkeit, Blinken/Nichtblinken. Dieser Speicherbereich wird durch ATTR(**Zahl 1**,**Zahl 2**) abgefragt.

Die **Zahl 1** gibt dabei die Zeile, **Zahl 2** die Bildschirmspalte des Zeichens wieder, das abgefragt werden soll.

Beispiel:

```
ATTR(12,21)
```

Anmerkungen: Beim Sinclair SPECTRUM kann entsprechend seiner Bildschirmaufteilung **Zahl 1** die Werte zwischen Ø und 23 und **Zahl 2** die Werte zwischen Ø und 32 annehmen.

Die einzelnen BITs werden beim SPECTRUM den Attributen wie folgt zugeordnet:

BIT 7	wenn 1	Zeichen blinkt
	wenn Ø	Zeichen blinkt nicht
BIT 6	wenn 1	helles Zeichen
	wenn Ø	dunkles Zeichen
BIT 3-5	Hintergrundfarbe	

BIT Ø-2	wenn ØØØ	schwarz
	wenn ØØ1	dunkelblau
	wenn Ø1Ø	rot
	wenn Ø11	violett
	wenn 1ØØ	grün
	wenn 1Ø1	hellblau
	wenn 11Ø	gelb
	wenn 111	weiß

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
› SINCLAIR SPECTRUM         :  ATTR (Zahl1,Zahl2)
```

AUTO

Befehlsart: Kommando

AUTO **Zahl 1, Zahl 2**

Beschreibung: Mit dem Kommando AUTO erfolgt die Numerierung beim Schreiben eines BASIC-Programmes automatisch. Nach jedem ENTER bzw. RETURN wird die neue Zeile mit der Darstellung ihrer Zeilennummer begonnen.

Zahl 1 gibt die Nummer der Zeile an, mit der die automatische Numerierung begonnen wird.

Zahl 2 gibt die Schrittfolge der Numerierung an.

AUTO 1ØØ, 1Ø

Hier beginnt die automatische Numerierung bei Zeilennummer 1ØØ und führt sie in Zehnerschritten fort.

AUTO

Das AUTO-Kommando kann man auch während der Eingabe eines Programmes verwenden, sofern **Zahl 1** größer ist als die Nummer der zuletzt eingegebenen Zeile.

Beispiel:

```
1 REM ----------------------
2 REM ----      DAS      ----
3 REM ----  GESPENSTER  ----
4 REM ----   PROGRAMM   ----
5 REM ----------------------


AUTO 10,10
```

Anmerkungen zur Kompatibilität:

Besonderheiten:

Commodore 16/116/128
Hier hat das Kommando das Format

AUTO **Zahl 2**

Es wird also nur die Schrittweite des AUTO-Befehls angegeben. Sobald man die erste Programmzeile eingegeben hat, wird automatisch in der angegebenen Schrittweite numeriert. Wir verlassen den Modus durch Eingabe einer Zeilennummer und -RETURN-.

Ausschalten läßt sich die automatische Zeilennumerierung auch, indem man AUTO ohne Parameter sowie RUN (siehe dort) eingibt.

Der Befehl kann nur im Direktmodus verwendet werden.

Besonderheiten:

Schneider CPC 464
Die automatische Zeilennumerierung kann durch Drücken der ESC-Taste ausgeschaltet werden.
Gerät die automatische Zeilennumerierung an Zeilennummern, zu denen schon Programmzeilen bestehen, so werden diese mit einem Stern gekennzeichnet. Hierdurch wird man vor einem Überschreiben gewarnt.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  AUTO Zahl2
> COMMODORE 128             :  AUTO Zahl2
> ENTERPRISE 64,128         :  AUTO AT Zahl1 STEP Zahl2
> MSX-STANDARD              :  AUTO Zahl1,Zahl2
> SCHNEIDER CPC464          :  AUTO Zahl1,Zahl2
> SCHNEIDER CPC664,CPC6128  :  AUTO Zahl1,Zahl2
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

BACKUP

Befehlsart: Kommando

BACKUP D**Zahl 1** TO D**Zahl 2** ON U**Zahl 3**
BACKUP D**Zahl 1** TO D**Zahl 2**, U**Zahl 3**

Beschreibung: Mit dem Befehl BACKUP wird eine Sicherheitskopie einer Diskette erstellt. Die Zieldiskette muß man dabei nicht formatieren, da der BACKUP-Befehl alle Informationen von der Quelldiskette auf die Zieldiskette überträgt. So erhält man eine Sicherheitskopie der Originaldiskette.
Dieser Befehl läßt sich nur bei zwei Laufwerken anwenden!
Die **Zahl 1** gibt das Laufwerk an, in dem sich die Originaldiskette, die sogenannte Quelldiskette, befindet. Es

sind also nur die Werte Ø oder 1 erlaubt. Die Laufwerksnummer ist jeweils auf dem Doppellaufwerk angegeben.

Die **Zahl 2** gibt das Laufwerk an, in dem sich die Kopie, die sogenannte Zieldiskette, befindet. Auch hier sind nur die Werte Ø und 1 erlaubt.

Die **Zahl 3** gibt die Geräteadresse des zweiten Laufwerkes an, wenn man mit zwei Einzellaufwerken operiert. Gewöhnlich handelt es sich dabei um die Zahlen 7, 9 oder 1Ø.

Bei Doppellaufwerken hat der Befehl das Format

BACKUP **DZahl 1** TO **DZahl 2**

Bei Einzellaufwerken hat der Befehl das Format

BACKUP **DZahl 1** TO **DZahl 2** ON **UZahl 3**

oder

BACKUP **DZahl 1** TO **DZahl 2, UZahl 3**

Beispiele:

BACKUP DØ TO D1

(kopiert bei einem Doppellaufwerk den Disketteninhalt von Laufwerk Ø nach Laufwerk 1)

BACKUP DØ TO D1, U9

(kopiert bei Einzellaufwerken den Disketteninhalt von Laufwerk Ø nach Laufwerk 1 mit der Geräteadresse 9)

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	BACKUP DZahl1 TO DZahl2 ON UZahl3
› COMMODORE 128	:	BACKUP DZahl1 TO DZahl2 ON UZahl3
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

BANK

Befehlsart: Anweisung

BANK **Zahl**

Beschreibung: Mit BANK wählen wir eine andere "Speicherbank" im Rechnerspeicher an.

Bei Befehlen wie PEEK, POKE, SYS oder WAIT (siehe jeweils dort!) können nur (im speziellen Fall) 65536 Speicherplätze (= 64 K-BYTE) angesprochen werden. Es gibt aber schon mehrere Computer, die einen größeren Speicher als 64 K-BYTE haben. Man kann nun mit dem BANK-Befehl die einzelnen Speicherbänke (jeweils 64 K-BYTE) ansprechen. Die **Zahl** gibt dabei die Speicherbanknummer an. Sie liegt bei 1MBYTE adressierbarem Speicher zwischen Ø und 15.

Beispiele:

```
10 INPUT "Speicherbank"; B
20 BANK B
```

```
10 FOR I=0 TO 65535
20 BANK1:POKE I,0
30 NEXT

READY.
```

Anmerkung zur Kompatibilität: Beim Commodore 128 lassen sich in der serienmäßigen Version drei verschiedene Speicherbänke ansprechen:

BANK Ø Speicher für BASIC-Text
BANK 1 Speicher für Variablen
BANK 15 feste Speicher für BASIC, Maschinensprachemonitor und Betriebssystem

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  BANK Zahl
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

BASE

Befehlsart: Funktion

BASE(**Zahl**) Abfrage der Tabelle
BASE(**Zahl**)=**Zahl 1** Beschreiben der Tabelle

Beschreibung: Mit dieser Funktion kann die VDP-Tabelle beschrieben oder gelesen werden.

VDP bedeutet VIDEO DISPLAY PROCESSOR. Dieser Rechner-Baustein ist für die Bildschirmdarstellungen verantwortlich.

Die **Zahl** liegt bei diesem Befehl zwischen Ø und 19, wobei allerdings die Zahlen 1, 3, 4 und 16 nicht verwendet werden. Sie gibt den Tabellenplatz an, der abgefragt werden soll.

Zahl 1 liegt zwischen Ø und 65535. Sie ist die "Adresse" beim Beschreiben der Tabelle.

Beispiel:

PRINT BASE(Ø)

Mit diesem Format wird die VDP-Tabelle abgefragt.

BASE(15)=2Ø48

Mit diesem Format wird der VDP-Tabelle ein Wert zugewiesen.

Anmerkung: Die Anwendung dieses Befehls setzt gute BASIC-Programmierkenntnisse voraus, vor allem auch Kenntnisse über den Aufbau des VDP-Bausteines.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	BASE(Zahl) BASE(Zahl)=Zahl1
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

BEEP

Befehlsart: Anweisung

BEEP **Zahl 1,Zahl 2**

Beschreibung: Es wird ein Ton erzeugt. **Zahl 1** gibt die Tondauer an, **Zahl 2** die Tonhöhe. Dabei sind Werte zwischen -6Ø und 69 erlaubt. Die Zahl Ø entspricht dem Ton C.

Beispiel:

```
10  tonleiter
20  "tondauer ",x
30  i=0  12
40  x,i
50  i

ready.
```

Anmerkung zur Kompatibilität: Bei MSX-Computern verwendet man den Befehl ohne Parameter, also

BEEP

Es wird ein kurzer Ton erzeugt, der sich allerdings nicht verändern läßt (siehe auch PING).

Zur Erzeugung von Musiknoten verwenden wir hier den SOUND-Befehl (siehe dort).

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :  PING
> MSX-STANDARD              :  BEEP
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
> SINCLAIR SPECTRUM         :  BEEP Zahl1,Zahl2
```

BEGIN...BEND

Befehlsart: Anweisung

BEGIN
BEND

Beschreibung: Mit diesem Befehl kann man beim IF...THEN... ELSE-Befehl (siehe IF...THEN) nach dem THEN-Befehl Anweisungen schreiben, die über eine Programmzeile hinausgehen.

Dabei beginnt man nach dem THEN mit dem BEGIN. Anschließend werden die Anweisungen angefügt. Den Programmteil schließen wir mit dem BEND-Befehl ab. Befinden sich in der Programmzeile, in der ein BEND erfolgte, noch Anweisungen, gehören sie zum IF...THEN...ELSE-Befehl.

Beispiel:

```
10 print"Bist du ein Junge (j/n)?"
20 get a$: if a$<>"j" and a$<>"n" then 20
30 if a$="n" then begin
40 print "Toll! Ich stehe auf"
50 print "Maedchen an meiner
60 print "Tastatur."
70 bend:else print "Lass doch mal ein Maedchen computern!"
80 end
```

Kompatibilitätsliste:

	ATARI 600,800,130XE	:		
	COMMODORE VC20,64	:		
	COMMODORE 16,116,PLUS4	:		
›	COMMODORE 128	:	BEGIN ...	BEND
	ENTERPRISE 64,128	:		
	MSX-STANDARD	:		
	SCHNEIDER CPC464	:		
	SCHNEIDER CPC664,CPC6128	:		
	SINCLAIR ZX81	:		
	SINCLAIR SPECTRUM	:		

BIN

Befehlsart: Funktion

BIN(Zahl)

Beschreibung: Die Funktion wandelt eine binäre Zahl in eine Dezimalzahl um.

Die Zahl bildet das binäre Argument. Es darf nur aus einer Kombination der Zahlen 1 und Ø bestehen und kann bis zu 1Ø Stellen umfassen. Zwar ist eine Eingabe binärer Zahlen mit mehr als 1Ø Stellen möglich, doch in diesem Fall werden die letzten Zahlen immer als Ø interpretiert.

Beispiele:

```
PRINT BIN (11Ø1)
 13
READY

PRINT BIN (11111111111)
 2Ø46
READY
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  BIN(Zahl)
  MSX STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
> SINCLAIR SPECTRUM         :  BIN Zahl
```

BIN$

Befehlsart: Funktion

BIN$(**Zahl 1**,**Zahl 2**)

Beschreibung: Diese Funktion gibt die **Zahl 1** als String wieder, der die binäre Darstellung (siehe Zahlensysteme) dieser Zahl enthält.

Die **Zahl 2** gibt die Länge des binären Strings wieder. Ist der String länger, als die **Zahl 2** angibt, wird sie ignoriert. Ist er kürzer, wird links die entsprechende Zahl von Nullen wiedergegeben. **Zahl 2** kann dabei einen Wert zwischen 1 und 16 annehmen.

PRINT BIN$(27,8) ergibt als Binärstring ØØØ11Ø11

Beispiel:

```
10 FOR I=1 TO 500
20 PRINT I;"  ALS BINAERZAHL: ";BIN$ (I,8)
30 PRINT:PRINT "NAECHSTE ZAHL: >N< DRUECKEN":PRINT
40 A$=INKEY$
50 IF A$="N" OR A$= "n" THEN 70
60 GOTO 40
70 NEXT I
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	BIN$(Zahl1)
› SCHNEIDER CPC464	:	BIN$(Zahl1,Zahl2)
› SCHNEIDER CPC664,CPC6128	:	BIN$(Zahl1,Zahl2)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

BLOAD (für Commodore 128)

Befehlsart: Kommando

BLOAD "**Programmname**",D**Zahl 1**,U**Zahl 2**,ON B**Zahl 3**,P**Adresse**

Beschreibung: Mit diesem Befehl kann man ein Maschinenprogramm in den Rechner laden und starten. Der **Programmname** gibt den Namen des Programms an, das in den Rechnerspeicher geladen werden soll. Dieser **Programmname** kann bis zu 16 Zeichen lang sein. **Zahl 1** gibt die Laufwerksnummer der Diskettenstation an, von der geladen werden soll. Ihr Wert beträgt entweder 1 oder Ø. **Zahl 1** kann auch entfallen.

Zahl 2 gibt die Geräteadresse an. Ihr Wert liegt zwischen 4 und 15. **Zahl 2** kann entfallen, wenn man nur ein Laufwerk oder ein Doppellaufwerk verwendet.

Zahl 3 gibt den Speicherbereich an, in den geladen werden soll. Der voreingestellte Wert ist dabei Ø. Er wird angenommen, wenn man keinen Wert für **Zahl 3** angibt.

Das Programm wird unter der **Adresse** abgelegt. Ihr Wert kann zwischen Ø und 65535 liegen. Wenn man die **Adresse** nicht angibt, wird die Adresse aus den ersten beiden Bytes des Programmes gebildet.

Beispiele:

```
BLOAD"TEST",DØ,U8,ONB1,P4ØØØØ
BLOAD"VERSUCH",U8
```

Anmerkung zur Kompatibilität: Der Befehl BLOAD hat im BASIC der MSX-Computer eine ähnliche Wirkung, aber ein anderes Format.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :
        BLOAD "Dateiname",DZahl1,UZahl2,ON BZahl3,PAdr
```

```
ENTERPRISE 64,128          :
MSX-STANDARD               :
      BLOAD "Dateiname",Zeichen,Adresse
SCHNEIDER CPC464           :
      LOAD "Dateiname",Adresse
SCHNEIDER CPC664,CPC6128   :
      LOAD "Dateiname.BIN",Adresse
SINCLAIR ZX81              :
SINCLAIR SPECTRUM          :
```

BLOAD (für MSX-Computer)

Befehlsart: Kommando

BLOAD "**Programmname**", **Zeichen**, **Adresse**

Beschreibung: Ein Maschinenprogramm wird in den Rechner geladen und gestartet.

Der **Programmname** besteht aus zwei Teilen, der Gerätebezeichnung und dem eigentlichen Programmnamen. Die Gerätebezeichnung gibt das Gerät an, von dem geladen werden soll (zum Beispiel CAS: entspricht dem Datenrekorder). Der Programmname gibt an, welches Programm geladen werden soll.

Bei **Zeichen** wird entweder ein R (= RUN) eingegeben – dann wird das Programm nach dem Laden gestartet –, oder es wird durch Eingabe eines Kommas übergangen.

Die **Adresse** kennzeichnen wir durch eine ganze Zahl. Sie wird zu der eigentlichen Programmadresse hinzugezählt, und an dem so entstandenen Adreß-Platz wird das Programm abgespeichert. Die Angabe der **Adresse** kann auch entfallen.

Beispiel:

```
BLOAD "CAS:TEST",R,12ØØØ
BLOAD "CAS:VERSUCH",1ØØØ
BLOAD "CAS:PROBE"
```

Anmerkung zur Kompatibilität: Der Befehl BLOAD hat im BASIC des Commodore 128 eine ähnliche Wirkung, aber ein anderes Format.

Kompatibilitätsliste:

```
ATARI 600,800,130XE       :
COMMODORE VC20,64         :
COMMODORE 16,116,PLUS4    :
COMMODORE 128             :
      BLOAD "Dateiname",DZahl1,UZahl2,ON BZahl3,PAdresse
ENTERPRISE 64,128         :
> MSX-STANDARD            :
      BLOAD "Dateiname",Zeichen,Adresse
SCHNEIDER CPC464          :
      LOAD "Dateiname",Adresse
SCHNEIDER CPC664,CPC6128  :
      LOAD "Dateiname.BIN",Adresse
SINCLAIR ZX81             :
SINCLAIR SPECTRUM         :
```

BOOT

Befehlsart: Kommando

BOOT "**Programmname**",D**Zahl 1**,U**Zahl 2**

Beschreibung: Mit diesem Befehl wird eine Binärdatei in den Rechnerspeicher geladen und, sofern sie ein Maschinenprogramm darstellt, auch gestartet.

Der **Programmname** der Datei, die geladen werden soll, darf höchstens 16 Zeichen lang sein.

Zahl 1 gibt die Nummer des Laufwerks an. Die Zahl kann die Werte Ø oder 1 annehmen. Sie darf (wie im Beispiel unten) auch fehlen.

Zahl 2 gibt die Geräteadresse an. Allerdings muß man die Geräteadresse nur angeben, falls sie aufgrund der Verwendung von zwei Einzellaufwerken benötigt wird. Sie liegt normalerweise zwischen 7 und 1Ø.

Beispiele:

BOOT "Hilfe"

BOOT "VERSUCH",DØ

BOOT "Test",DØ,U8

Anmerkung: Gibt man BOOT ohne Parameter an, so wird aus dem Floppylaufwerk, dem die Geräteadresse 8 zugewiesen wurde, automatisch vom Betriebssystem jene Datei geladen, die auf der Diskette im Sektor Ø der Spur 1 abgelegt ist.
Dies ist zum Beispiel der Fall, wenn man CP/M von der Diskette lädt.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  BOOT "Programmname",DZahl1,UZahl2
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

BORDER

Befehlsart: Anweisung

BORDER **Zahl 1, Zahl 2**

Beschreibung: Die Randfarbe des Bildschirmes wird bestimmt.
Die **Zahl 1** gibt dabei die gewünschte Randfarbe an. Wird eine **Zahl 2** angegeben, so blinkt der Rand zwischen den Farbwerten von **Zahl 1** und **Zahl 2.** Dabei läßt sich mit dem Befehl SPEED INK (siehe dort) die Blinkfrequenz festlegen.

BORDER 1,13

läßt die Randfarbe des Bildschirmes zwischen Schwarz und Weiß blinken.

Beispiel:

```
10 INPUT "DEIN NAME";NA$
20 CLS
30 FOR I = 1 TO 25
40 PRINT "          WILLKOMMEN, ";NA$;"!"
50 BORDER 1,13
60 NEXT I
70 BORDER 1
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :  SETCOLOR 4,Zahl1,8
  COMMODORE 64              :  POKE 53280,Zahl1
  COMMODORE 16,116,PLUS4    :  COLOR 4,Zahl1
  COMMODORE 128             :  COLOR 4,Zahl1
  ENTERPRISE 64,128         :  SET BORDER Zahl1
  MSX-STANDARD              :  COLOR ,,Zahl1
> SCHNEIDER CPC464          :  BORDER Zahl1,Zahl2
> SCHNEIDER CPC664,CPC6128  :  BORDER Zahl1,Zahl2
  SINCLAIR ZX81             :
> SINCLAIR SPECTRUM         :  BORDER Zahl1
```

BOX

Befehlsart: Anweisung

BOX Zahl 1,Zahl 2,Zahl 3,Zahl 4,Zahl 5,Zahl 6,Zahl 7

Beschreibung: Auf dem Bildschirm wird ein Rechteck gezeichnet. Dieses läßt sich drehen und ausmalen.

Der Befehl setzt voraus, daß man mit GRAPHIC (siehe dort) einen Grafik-Bildschirm definiert hat.

Zahl 1 gibt die Farbquelle an. Diese Farbquelle enthält die Farbwerte, die man mit Hilfe der COLOR-Anweisung (sie-

he dort) definiert hat. Die **Zahl 1** liegt zwischen Ø und 3.

Die Zahlen **Zahl 2** bis **Zahl 5** bestimmen die Koordinaten des Rechtecks. **Zahl 2** und **Zahl 4** geben die Grenzen in X-Richtung (Breite des Rechtecks) an, die anderen beiden Zahlen die Grenzen in Y-Richtung (Höhe des Rechtecks).

Mit **Zahl 6** kann man einen Winkel in Grad angeben, um den das Rechteck gedreht werden soll. Dabei entstehen Parallelogramme. Der Wert dieser Zahl bewegt sich zwischen Ø und 36Ø.

Mit **Zahl 7** läßt sich angeben, ob das Rechteck ausgemalt werden soll (**Zahl 7**=1) oder nicht (**Zahl 7**=Ø).

Beispiel:

```
10 REM MODERNE KUNST
20 GRAPHIC 3,1                  :REM MEHRFARBENGRAFIK EINSCHALTEN
30 COLOR 0,1                    :REM SCHWARZER HINTERGRUND
40 COLOR 1,2                    :REM WEISSER VORDERGRUND
50 COLOR 2,8                    :REM GELBE MEHRFARBE1
60 COLOR 3,7                    :REM BLAUE MEHRFARBE2
70 Z1=INT(RND(1)*4)             :REM ZUFALLSFARBQUELLE
80 Z2=INT(RND(1)*160)           :REM ZUFALLS X-KOORDINATE1
90 Z3=INT(RND(1)*200)           :REM ZUFALLS Y-KOORDINATE1
100 Z4=INT(RND(1)*160)          :REM ZUFALLS X-KOORDINATE2
110 Z5=INT(RND(1)*200)          :REM ZUFALLS Y-KOORDINATE2
120 Z6=INT(RND(1)*360)          :REM ZUFALLS DREHWINKEL
130 Z7=INT(RND(1)*2)            :REM ZUFAELLIGES AUSMALEN
140 BOX Z1,Z2,Z3,Z4,Z5,Z6,Z7
150 GOTO 70
```

Das Programm malt zufällige Rechtecke auf den Bildschirm.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :
    BOX Zahl1,Zahl2,Zahl3,Zahl4,Zahl5,Zahl6,Zahl7
> COMMODORE 128             :
    BOX Zahl1,Zahl2,Zahl3,Zahl4,Zahl5,Zahl6,Zahl7
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

BRIGHT

Befehlsart: Anweisung

BRIGHT **Zahl**

Beschreibung: Mit diesem Befehl kann man Zeichen durch Verändern der Helligkeit gegenüber anderen Stellen des Bildschirms hervorheben. Diese Veränderung bleibt so lange erhalten, bis ein neuer BRIGHT-Befehl ausgeführt wird.

Die **Zahl** gibt die Veränderung an. Sie kann nur die Werte 1 oder Ø annehmen.

Ø normale Helligkeit
1 gesteigerte Helligkeit

Beispiel:

```
10 PRINT "COMPUTER"
20 PRINT"BUECHER"
30 BRIGHT 1
40 PRINT"SCHNEIDER"
50 BRIGHT 0
60 FOR T=1 TO 500:NEXT
70 GOTO 10
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
> SINCLAIR SPECTRUM         :  BRIGHT Zahl
```

BSAVE (für Commodore 128)

Befehlsart: Kommando

BSAVE "**Programmname**",D**Zahl 1**,U**Zahl 2**,ON BZ**ahl 3**,P**Adresse 1** TO P**Adresse 2**

Beschreibung: Mit diesem Befehl kann man einen bestimmten Bereich eines Maschinensprache-Programmes abspeichern.

Der **Programmname**, unter der die Datei bzw. das Programm abgespeichert werden soll, darf bis zu 16 Zeichen lang sein.

Zahl 1 gibt die Laufwerksnummer der Diskettenstation an, auf der es gespeichert werden soll. Der Wert kann Ø oder 1 betragen. Diese Zahl kann auch entfallen.

Zahl 2 nennt die Geräteadresse. Ihr Wert liegt zwischen 4 und 15. Sie kann entfallen, wenn man nur ein Laufwerk oder ein Doppellaufwerk verwendet.

Zahl 3 gibt die Speicherbank an (siehe auch unter BANK), von der das Programm abgespeichert werden soll. Der Wert liegt zwischen Ø und 15. Den voreingestellten Wert Ø verwendet man, wenn kein eigener Wert angegeben wird.

Adresse 1 ist die Startadresse und **Adresse 2** die Endadresse des Speicherbereiches, der abgespeichert werden soll.

Beispiel:

```
BSAVE"TEST",DØ,U8,ONB1,P4ØØØ TO P5ØØØ
BSAVE "VERSUCH", P1Ø24 TO P4Ø96
```

Anmerkung: Mit "Laufwerk" wird eine Diskettenstation bezeichnet. Mit "Gerät" bezeichnet man Peripheriegeräte wie Drucker, Diskettenstation, Datenrekorder oder Plotter.

Anmerkung zur Kompatibilität: Der Befehl BSAVE hat im MSX-BASIC eine ähnliche Wirkung, aber ein anderes Format (siehe dort).

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :
        BSAVE "Dateiname",DZahl1,UZahl2,ON BZahl3,PAdresse1 TO PAdresse2
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
        BSAVE "Dateiname",Adresse1,Adresse2,Adresse3
  SCHNEIDER CPC464          :
        SAVE "Dateiname",B,Adresse1,(Adresse2-Adresse1)
  SCHNEIDER CPC664,CPC6128  :
        SAVE "Dateiname.BIN",Adresse1,(Adresse2-Adresse1),Adresse3
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

BSAVE (für MSX-Computer)

Befehlsart: Kommando

BSAVE "**Programmname**",**Adresse 1**,**Adresse 2**,**Adresse 3**

Beschreibung: Ein bestimmter Bereich eines Maschinenprogrammes wird abgespeichert.

Der **Programmname** besteht aus zwei Teilen, der Gerätebezeichnung und dem eigentlichen Programmnamen. Die Gerätebezeichnung gibt das Gerät an, auf der es gespeichert werden soll (z. B. CAS:). Der **Programmname** bezeichnet das Programm, das abgespeichert und später mit BLOAD (siehe dort) wieder geladen werden kann.

Adresse 1 ist die Startadresse und **Adresse 2** die Endadresse des Speicherbereiches, der abgespeichert werden soll.

Adresse 3 gibt die spätere Startadresse für das Programm an, wenn es in den Rechner geladen werden soll. Gibt man hierfür keinen Wert ein, wird automatisch **Adresse 1** als spätere Startadresse angenommen.

Beispiele:

```
BSAVE "CAS:TEST",&H1ØØØ,&H2ØØØ,&H2ØØØ
BSAVE "CAS:VERSUCH",1Ø24,4Ø96
```

Anmerkung zur Kompatibilität: Der Befehl BSAVE hat im BASIC des Commodore 128 eine ähnliche Wirkung, aber ein anderes Format.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
        BSAVE "Dateiname",DZahl1,UZahl2,ON BZahl3,PAdresse1 TO PAdresse2
  ENTERPRISE 64,128         :
> MSX-STANDARD              :
        BSAVE "Dateiname",Adresse1,Adresse2,Adresse3
  SCHNEIDER CPC464          :
        SAVE "Dateiname",B,Adresse1,(Adresse2-Adresse1)
  SCHNEIDER CPC664,CPC6128  :
        SAVE "Dateiname.BIN",Adresse1,(Adresse2-Adresse1),Adresse3
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

BUMP

Befehlsart: Funktion

BUMP(**Zahl**)

Beschreibung: Die Funktion BUMP gibt die Nummer eines Sprites zurück, das zuletzt mit einem anderen Sprite oder Zeichen auf dem Bildschirm zusammengestoßen ist.

Die **Zahl** gibt dabei an, womit das Sprite zusammengestoßen ist. Beträgt die **Zahl** 1, wird nach einem Zusammenstoß von Sprite und Sprite gefragt. Beträgt die **Zahl** 2, wird nach einem Zusammenstoß von Sprite und Zeichen gefragt.

Der Wert der zurückgegebenen Zahl liegt zwischen Ø und 8. Ø bedeutet, daß kein Zusammenstoß stattgefunden hat. Die Werte 1 bis 8 bezeichnen die Nummer des Sprites, das kollidiert ist.

Nach erfolgter Abfrage wird BUMP auf Ø zurückgesetzt.

Beispiele:

I=BUMP (1)

PRINT BUMP (2)

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
› COMMODORE 128	:	BUMP(Zahl)
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

BYE

Befehlsart: Kommando

BYE

Beschreibung: Nach Eingabe und Bestätigung von BYE im Direktmodus wird BASIC verlassen, und es wird ein Menü zur Überprüfung der internen Funktionen des Computers aufgerufen. Im Rechner vorhandene Programme gehen dabei verloren.

Es stehen drei Befehle zur Verfügung. Man wählt sie durch Drücken der entsprechenden Tasten aus:

SELECT wählt im Testmenü jenen Programmpunkt aus, der angibt, welche Funktion überprüft werden soll.

START Der durch Blinken angezeigte Menüteil wird gestartet.

RESET veranlaßt die Rückkehr in den BASIC-Modus. Der Rechner ist dabei "resetet" (siehe RESET) worden, also "zurückgestellt".

Das Menü weist vier Teile auf:

MEMORY
Dieser Teil überprüft den Speicher im Computer. Sind die Bausteine in Ordnung, so werden sie in grüner Farbe dargestellt.

AUDIO-VISUAL
Dieser Teil überprüft die Tonkanäle auf ihre Funktion. Dabei werden alle vier Tonkanäle getestet.

KEYBOARD TEST
Dieser Teil überprüft, ob die Tastatur des Computers voll funktionstüchtig ist. Bei Drücken der Tasten leuchten die entsprechenden Tasten im Bildschirm grün auf.

ALL TESTS
Mit diesem Teil werden alle drei genannten Tests der Reihe nach durchgeführt.

Mit der HELP-Taste (siehe HELP) kehrt man in das Menü zurück.

Beispiel:

```
BYE
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	: BYE
COMMODORE VC20,64	:
COMMODORE 16,116,PLUS4	:
COMMODORE 128	:
ENTERPRISE 64,128	:
MSX-STANDARD	:
SCHNEIDER CPC464	:
SCHNEIDER CPC664,CPC6128	:
SINCLAIR ZX81	:
SINCLAIR SPECTRUM	:

CALL

Befehlsart: Anweisung

CALL **Adresse, Parameterliste**

Beschreibung: CALL ruft ein Unterprogramm auf, das in Maschinensprache geschrieben wurde. Die Adresse wird dabei durch **Adresse** festgelegt. Zusätzlich hat man die Möglichkeit, durch die **Parameterliste** Zahlenwerte zu übergeben, die in der Maschinenroutine verwendet werden.

Beispiel:

```
CALL &BD19
```

Anmerkung: Nur wer schon viel Erfahrung in der Programmierung gesammelt hat, sollte diesen Befehl verwenden. In jedem Fall empfiehlt es sich, das Programm bei Verwendung des CALL-Befehls abzuspeichern, bevor es gestartet wird. Bei falscher **Adresse** kann der Rechner "abstürzen".

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :  USR(Adresse,Parameterliste)
  COMMODORE VC20,64         :  SYS Adresse,Parameterliste
  COMMODORE 16,116,PLUS4    :  SYS Adresse,Parameterliste
  COMMODORE 128             :  SYS Adresse,Parameterliste
> ENTERPRISE 64,128         :  CALL Programmname(Parameterliste)
> MSX-STANDARD              :  CALL Funktionsname(Parameterliste)
/ SCHNEIDER CPC464          :  CALL Adresse,Parameterliste
> SCHNEIDER CPC664,CPC6128 :  CALL Adresse,Parameterliste
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

CAPTURE

Befehlsart: Kommando

CAPTURE FROM #**Kanal 1** TO #**Kanal 2**

Beschreibung: CAPTURE bewirkt, daß **Kanal 1** gegenüber **Kanal 2** beim Einlesen und Bearbeiten von Daten der Vorzug gegeben wird. Das heißt, Daten die in **Kanal 1** anstehen, werden zuerst eingelesen. Die Werte für die Kanalnummern von **Kanal 1** und **Kanal 2** können dabei zwischen 0 und 255 liegen.

Beispiel:

```
CAPTURE FROM #1 TO #5
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE        :
  COMMODORE VC20,64          :
  COMMODORE 16,116,PLUS4     :
  COMMODORE 128              :
> ENTERPRISE 64,128          :  CAPTURE FROM #Kanal1 TO #Kanal2
  MSX-STANDARD               :
  SCHNEIDER CPC464           :
  SCHNEIDER CPC664,CPC6128   :
  SINCLAIR ZX81              :
  SINCLAIR SPECTRUM          :
```

CASE

Befehlsart: Anweisung

Die Wirkung der Anweisung CASE ist identisch mit der von SELECT (siehe dort).

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  siehe SELECT
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

CAT

Befehlsart: Kommando

CAT

Beschreibung: Das Kommando CAT erstellt das Inhaltsverzeichnis einer Kassette oder einer Diskette.

Beispiel:

CAT
CAT2
CAT#4;1

Anmerkungen zur Kompatibilität:

Besonderheiten:

Schneider CPC
Auf dem Bildschirm wird das Inhaltsverzeichnis in folgender Form wiedergegeben (Kassette):

Dateiname Blocknummer Dateikennzeichen OK

Dabei bedeutet:

Dateiname	Name der Datei oder des Programmes
Blocknummer	Die Anzahl der Blocks gibt die Länge eines Programmes wieder
Dateikennzeichen	Das Dateikennzeichen gibt die Art einer Datei bzw. eines Programmes wieder. ($ = BASIC-Programm, % = geschütztes BASIC-Programm; * = ASCII-Testdatei; & = binäre Datei in Maschinensprache)
OK	Der genannte Block einer Datei bzw. eines Programmes ist zu Ende

Beim CATalogisieren einer Diskette wird auch der noch freie Speicherplatz auf der Diskette wiedergegeben.

Sinclair ZX Spectrum

Das Kommando hat hier das Format

CAT **Zahl 1;Zahl 2**

Zahl 1 ist die Dateinummer. Sie kann zwischen Ø und 15 liegen. **Zahl 2** ist ein numerischer Ausdruck (Werte zwischen 1 und 8), der die Nummer des Microdrives angibt.

Kompatibilitätsliste:

ATARI 600,800;130XE	:		
COMMODORE VC20,64	:		
COMMODORE 16,116,PLUS4	:	DIRECTORY	Diskette
COMMODORE 128	:	CATALOG oder DIRECTORY	Diskette
ENTERPRISE 64,128	:		
MSX-STANDARD	:		
› SCHNEIDER CPC464	:	CAT	Kassette
› SCHNEIDER CPC664,CPC6128	:	CAT	Diskette
SINCLAIR ZX81	:		
› SINCLAIR SPECTRUM	:	CAT Laufwerksnummer	Diskette

CATALOG

Befehlsart: Kommando

CATALOG
CATALOG DZahl 1,UZahl 2,"Programmname"

Beschreibung: Das Kommando gibt das Inhaltsverzeichnis der durch **Zahl 1** (Laufwerknummer) oder **Zahl 2** (Geräteadresse) bestimmten Diskette wieder. Der Befehl ist in seiner Wirkung identisch mit DIRECTORY (siehe dort).

Kompatibilitätsliste:

ATARI 600,800,130XE	:		
COMMODORE VC20,64	:		
COMMODORE 16,116,PLUS4	:	DIRECTORY	Diskette
› COMMODORE 128	:	CATALOG oder DIRECTORY	Diskette
ENTERPRISE 64,128	:		
MSX-STANDARD	:		
SCHNEIDER CPC464	:	CAT	Kassette
SCHNEIDER CPC664,CPC6128	:	CAT	Diskette
SINCLAIR ZX81	:		
SINCLAIR SPECTRUM	:	CAT Laufwerksnummer	Diskette

CAUSE EXEPTION

Befehlsart: Anweisung

CAUSE EXCEPTION Zahl

Beschreibung: Die Anweisung dient dem Benutzer dazu, eigene Fehlermeldungen zu definieren.

Die Zahl gibt die Nummer des Fehlers an. Sie sollte zwischen Ø und 999 liegen, denn diese Fehlernummern werden nicht von BASIC benutzt. Größere Werte für die Zahl dagegen sind BASIC-Fehlermeldungen.

Im Falle des ausgeführten WHEN EXCEPTION (siehe dort) wird in das dort definierte Unterprogramm verzweigt. An-

dernfalls wird das Programm mit Fehlermeldung unterbrochen.

Beispiel:

```
10 REM
20 WHEN EXCEPTION USE FEHLER
30 PRINT "GEBE EINE ZAHL EIN"
40 INPUT I
50 IF I>10 THEN CAUSE EXCEPTION 1
60 PRINT "WURZEL AUS"I"GLEICH"SQR(I)
70 IF I<>0 THEN 30
80 END WHEN
90 PRINT "PROGRAMMENDE
100 END
1000 HANDLER FEHLER
1010    IF EXTYPE=1 THEN PRINT "MIT SO GROSSEN ZAHLEN RECHNE ICH NICHT":I=1
1020    IF EXTYPE>999 THEN PRINT EXSTRINGS(EXTYPE):PRINT "IN ZEILE"EXLINE:I=1
1030    RETRY
1040 END HANDLER
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	CAUSE EXCEPTION Zahl
MSX-STANDARD	:	ERROR Zahl
SCHNEIDER CPC464	:	ERROR Zahl
SCHNEIDER CPC664,CPC6128	:	ERROR Zahl
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

CDBL

Befehlsart: Funktion

CDBL(**Zahl 1**)

Beschreibung: Eine Zahl von einfacher Genauigkeit (6 Stellen) wird in eine Zahl mit doppelter Genauigkeit (14 Stellen) umgewandelt.

Die **Zahl** kann ein fester Wert wie .23445654 sein. Sie kann aber auch durch eine definierte Variable (wie zum

Beispiel X) oder einen Rechenausdruck vertreten sein (z. B. 5/3).

Beispiele:

```
PRINT CDBL(5/3)
 1.6666666666666
READY
```

Bei

```
10 ...
20 a# =CDBL(b!)
30 ...
```

entspricht a# der Zahl mit doppelter Genauigkeit; b! entspricht der Zahl mit einfacher Genauigkeit.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	CDBL(Zahl)
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

CEIL

Befehlsart: Funktion

CEIL(**Zahl**)

Beschreibung: Die **Zahl** wird auf den nächstgrößeren ganzen Wert aufgerundet.

Beispiele:

```
PRINT CEIL(2.3)
 3
READY

A=-1.23:B=CEIL(A):PRINT B
-1
READY
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	CEIL(Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

CHAIN

Befehlsart: Kommando

CHAIN "**Dateiname**",**Zeilennummer**

Beschreibung: Ein Programm wird von Kassette oder Diskette geladen und bei einer bestimmten **Zeilennummer** gestartet. Das im Speicher befindliche Programm wird dabei ersetzt. Ist keine **Zeilennummer** angegeben, wird das neue Programm von Anfang an ausgeführt.

Bei Verwendung einer Kassette kann der Befehl das Format CHAIN
haben. Dann wird das erste auf der Kassette gefundene Programm geladen. Bei Verwendung einer Diskette muß der **Dateiname** allerdings immer angegeben werden.

Mit CHAIN lassen sich auch Programme laden, die mit Hilfe von

SAVE "Dateiname",P

listgeschützt sind.

Beispiel:

```
10 PRINT "aaaaaa": GOTO 10
```

```
Abspeichern unter -a-
```

```
NEW
```

Anmerkung:

Mit

CHAIN "!**Dateiname**"

kann man die Meldungen des Computers zum Ladevorgang unterdrücken.

Beispiel:

```
10 PRINT "bbbbbb"
20 CHAIN "!a"
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE        :
  COMMODORE VC20,64          :
  COMMODORE 16,116,PLUS4     :
  COMMODORE 128              :
  ENTERPRISE 64,128          :
  MSX-STANDARD               :
> SCHNEIDER CPC464           :  CHAIN "Dateiname",Zeilennummer
> SCHNEIDER CPC664,CPC6128   :  CHAIN "Dateiname",Zeilennummer
  SINCLAIR ZX81              :
  SINCLAIR SPECTRUM          :
```

CHAIN MERGE

Befehlsart: Kommando

CHAIN MERGE "**Dateiname**", **Zeilennummer**, DELETE **Zeilennummer 2-Zeilennummer 3**

Beschreibung: Mit diesem Kommando läßt sich ein BASIC-Programm laden und zu einem bereits im Speicher befindlichen Programm hinzufügen.

Dabei ist zu beachten:

Enthält das dazugeladene Programm gleiche Zeilennummern wie das erste Programm, werden die Zeilen des ersten Programmes überschrieben. Alle anderen Zeilen bleiben erhalten, auch wenn es sich dabei um Zeilen mit höheren Nummern handelt, als das dazugeladene Programm aufweist.

Programme, die mit
SAVE"**Dateiname**,p"
gesichert wurden, kann man mit CHAIN MERGE nicht zusammenfügen.

Das sich neu ergebende Programm beginnt mit der Ausführung in jener Zeile, die durch die **Zeilennummer** angegeben wurde. Falls keine **Zeilennummer** angegeben wurde, beginnt es mit der ersten Zeile des Programmes. (Wichtig: Ist DELETE nicht angefügt, müssen auch in diesem Fall beide Kommata geschrieben werden!)

Um Teile des ursprünglichen Programmes zu löschen, setzt man den DELETE-Befehl ein. **Zeilennummer 2** bezeichnet den

Beginn, **Zeilennummer 3** das Ende des zu löschenden Programmteils.

ON ERROR GOTO, DEFINT, DEFREAL, DEFSTR sowie offene Dateien und Funktionen, die mit DEF FN definiert wurden, "vergißt" der Computer. Außerdem wird ein RESTORE durchgeführt, wenn das neue Gesamtprogramm nicht von der niedrigsten Zeile an abgearbeitet wird.

Bei Verwendung einer Kassette kann der **Dateiname** fortgelassen werden. (CHAIN MERGE ""). In diesem Fall wird das erste auf der Kassette gefundene Programm geladen. Bei Verwendung einer Diskette ist die Angabe des **Dateinamens** unerläßlich.

Fehlen bei DELETE der Bindestrich (-) und die **Zeilennummer 3**, so wird nur die Programmzeile mit **Zeilennummer 2** gelöscht.

Beispiel:

```
100 PRINT "aaaa"
110 PRINT "bbbb"
```

```
unter -a- abspeichern
```

```
NEW

10 PRINT "Anfang"
20 CHAIN MERGE "!a", 110, DELETE 20
200 PRINT "Ende"

run

Anfang
bbbb
Ende
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :
            CHAIN MERGE "Dateiname",Startzeile,DELETEZeilenbereich
> SCHNEIDER CPC664,CPC6128  :
            CHAIN MERGE "Dateiname",Startzeile,DELETEZeilenbereich
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

CHAR

Befehlsart: Anweisung

CHAR **Zahl 1,Zahl 2,Zahl 3,"Textausdruck",Zahl 4**

Beschreibung: Ein **Textausdruck** wird auf einer bestimmten Stelle auf den Bildschirm ausgegeben. Es spielt dabei keine Rolle, ob ein Textbildschirm oder ein Grafikbildschirm eingestellt ist.

Die **Zahl 1** gibt die Nummer der Farbquelle an. Sie kann Werte zwischen Ø und 3 annehmen. Die Farbquelle muß vorher mit dem COLOR-Befehl (siehe dort) festgelegt werden.

Die **Zahl 2** und die **Zahl 3** geben die Anfangsstelle des anzuzeigenden **Textausdrucks** an. Hierbei kennzeichnet **Zahl 2** die x-Koordinate, also die Spaltenposition (Ø-39 bei 4Ø Zeichen, Ø-79 bei 8Ø Zeichen); die **Zahl 3** gibt die y-Koordinate, also die Zeilenposition (Ø-24) an.

Die **Zahl 4** legt fest, ob der Text in normaler Darstellung (Ø) oder in reverser Darstellung (1) ausgegeben werden soll.

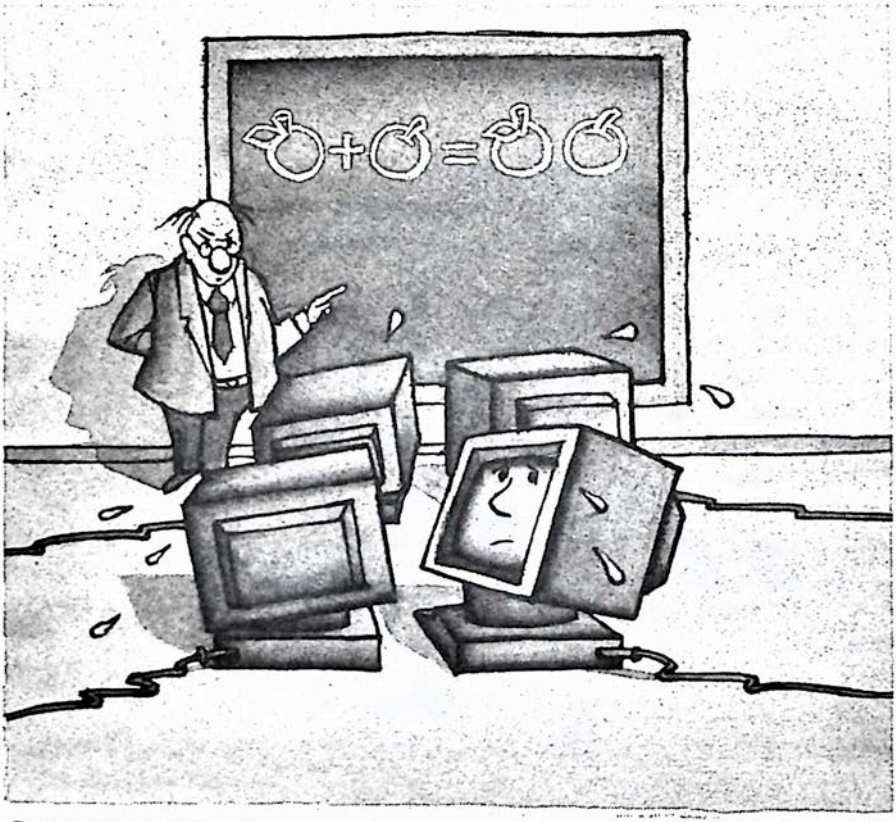
CHAR

Beispiele:

```
CHAR 1,12,13,"ICH BIN EIN COMMODORE"

CHAR 1,1,1,"ICH BIN MUEDE",1
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  CHAR Zahl1,Zahl2,Zahl3,"TEXT",Zahl4
> COMMODORE 128             :  CHAR Zahl1,Zahl2,Zahl3,"TEXT",Zahl4
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :  PRINT AT Zahl3,Zahl2;"Text"
  SINCLAIR SPECTRUM         :  PRINT AT Zahl3,Zahl2;"Text"
```



Computer können nur das, was ihnen der Mensch zuvor beigebracht hat.

CHR$

Befehlsart: Funktion

CHR$(**Zahl**)

Beschreibung: Mit dieser Funktion erhält man das zu der **Zahl** gehörige Zeichen, das ihr im ASCII-Code zugeordnet ist. Sie ist die Umkehrfunktion zu ASC (siehe dort!).

Ein Computer "weiß" nicht, was Buchstaben sind. Deshalb werden den Buchstaben Zahlen zugeordnet. Mit ihnen bestimmt er die Adresse, unter der er das Erscheinungsbild der Buchstaben abgespeichert hat. Eine solche Zahlenzuordnung nennt man einen Code. Der Code, der für die Computer verwendet wird, ist der ASCII-Code.

Beispiele:

```
PRINT CHR$ (64)
A
```

```
4 FOR T= 1 TO 20
5 FOR K = 1 TO 2
10 FOR I = 100 TO 200 STEP 0.05
20 PRINT CHR$(I);
30 NEXT I
35 POKE 53281,T
36 NEXT K
37 PRINT "[illegible]"
39 PRINT "[illegible]"
40 NEXT T
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	CHR$(Zahl)
› COMMODORE VC20,64	:	CHR$(Zahl)
› COMMODORE 16,116,PLUS4	:	CHR$(Zahl)
› COMMODORE 128	:	CHR$(Zahl)
› ENTERPRISE 64,128	:	CHR$(Zahl)
› MSX-STANDARD	:	CHR$(Zahl)
› SCHNEIDER CPC464	:	CHR$(Zahl)
› SCHNEIDER CPC664,CPC6128	:	CHR$(Zahl)
› SINCLAIR ZX81	:	CHR$ Zahl
› SINCLAIR SPECTRUM	:	CHR$ Zahl

CINT

Befehlsart: Funktion

CINT (**Zahl**)

Beschreibung: Die **Zahl** wird zu einer ganzen Zahl gerundet. Dabei gilt folgende Rundungsregel:

für positive Zahlen:
Zahl kleiner --.5 ergibt die nächstkleinere ganze Zahl
Zahl größer oder gleich --.5 ergibt die nächstgrößere ganze Zahl

für negative Zahlen:
Zahl kleiner oder gleich --.5 ergibt die nächstkleinere Zahl
Zahl größer als --.5 ergibt die nächstgrößere ganze Zahl
Dabei ist nicht zu vergessen: -1 ist größer als -2

Beispiel:

```
PRINT CINT (1.49)
 1

10 FOR x=0.01 TO 0.59 STEP 0.01
20 PRINT CINT(x);
30 NEXT x
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	CINT(Zahl)
› SCHNEIDER CPC664,CPC6128	:	CINT(Zahl)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

CIRCLE (für Commodore)

Befehlsart: Anweisung

CIRCLE **Zahl 1,Zahl 2,Zahl 3,...Zahl 9**

Beschreibung: Mit diesem Befehl kann man einen Kreis auf dem Bildschirm darstellen.

Zahl 1 bezeichnet die Farbquelle (siehe COLOR). Der Wert kann zwischen Ø und 3 liegen.
Zahl 2 bezeichnet die x-Achse (Ordinate) des Kreismittelpunktes in bezug zum Koordinaten-Ursprung (Nullpunkt des Koordinatensystems).
Zahl 3 bezeichnet die y-Achse (Abszisse).
Zahl 4 gibt den Radius des Kreises in x-Richtung an.
Zahl 5 gibt den Radius in y-Richtung an.
Zahl 6 ist der Startwinkel (Werte zwischen Ø und 36Ø).
Zahl 7 ist der Endwinkel (Werte zwischen Ø und 36Ø).
Zahl 8 kennzeichnet den Winkel, um den die Figur gedreht werden soll (Werte zwischen Ø und 36Ø).
Zahl 9 gibt den Segmentwinkel an (Werte zwischen 1 und 18Ø). Mit diesem Parameter kann man ein Vieleck zeichnen. Die **Zahl 9** läßt sich hierbei wie folgt berechnen:

Zahl 9 = 36Ø/(Anzahl der Ecken)

Um ein Viereck bzw. ein Quadrat zu bekommen, müßte man für die **Zahl 9** den Wert 9Ø eingeben.

Die Zahlen **Zahl 5, Zahl 6, Zahl 7, Zahl 8, Zahl 9** können jeweils entfallen. Wird eine der Zahlen ausgelassen, folgen im Format des Befehls zwei Kommata aufeinander.

Beispiel:

```
10 GRAPHIC1,1
20 FOR I=1 TO 15
30 CIRCLE1,100,100,I*4
40 NEXT I
50 GOTO 50
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :
        CIRCLE Zahl1,Zahl2,Zahl3,Zahl4,Zahl5,Zahl6,Zahl7,Zahl8,Zahl9
> COMMODORE 128             :
        CIRCLE Zahl1,Zahl2,Zahl3,Zahl4,Zahl5,Zahl6,Zahl7,Zahl8,Zahl9
  ENTERPRISE 64,128         :
        PLOT Zahl2,Zahl3, ELLIPSE Zahl4,Zahl5
  MSX-STANDARD              :
        CIRCLE STEP (Zahl2,Zahl3),Zahl4,Zahl1,Zahl6,Zahl5,(Zahl4/Zahl5)
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

CIRCLE (für MSX-Rechner)

Befehlsart: Anweisung

CIRCLE STEP(**Zahl 2,Zahl 3**),**Zahl 1,Zahl 6,Zahl 7,Zahl 11**

Beschreibung: Der MSX-Befehl CIRCLE ähnelt dem CIRCLE-Befehl des Commodore 128 (siehe dort). Das Format des Befehls weist allerdings kleine Unterschiede auf (zum besseren Vergleich wurden die Zahlen so numeriert, wie sie auch im Format des CIRCLE-Befehls beim Commodore 128 aufscheinen).

Mit dem MSX-Befehl CIRCLE lassen sich nicht nur Kreise, sondern auch Ellipsen auf den Bildschirm bringen.

Zahl 1 kennzeichnet die Farbnummer. Ihr Wert liegt zwischen Ø und 15.

Zahl 2 gibt die x-Richtung des Kreismittelpunktes im Verhältnis zum Koordinatenursprung an. Bei Angabe von STEP wird die vorherige Stellung des Grafik-Cursors zum Ursprung. STEP muß demgemäß entfallen, falls der normale Koordinatenursprung verwendet werden soll.

Zahl 3 gibt die y-Richtung an.

Zahl 6 gibt den Startwinkel an. Der Wert dafür liegt zwischen -2PI und 2PI (siehe auch unter PI).

Zahl 7 gibt den Endwinkel an. Der Wert liegt wiederum zwischen -2PI und 2PI.

Zahl 11 gibt das "Schlankheitsverhältnis" für einen Kreis an, der damit zur Ellipse wird.

Die Zahlen **Zahl 6**, **Zahl 7** und **Zahl 11** können jeweils entfallen. Wird eine der Zahlen ausgelassen, folgen im Format des Befehls zwei Kommata aufeinander.

Beispiel:

```
10 SCREEN 2
20 FOR I= 1 TO 15
30 CIRCLE(100,100),I*4,I,,,0.75
40 NEXT I
50 GOTO 50
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
        CIRCLE Zahl1,Zahl2,Zahl3,Zahl4,Zahl5,Zahl6,Zahl7,Zahl8,Zahl9
  COMMODORE 128             :
        CIRCLE Zahl1,Zahl2,Zahl3,Zahl4,Zahl5,Zahl6,Zahl7,Zahl8,Zahl9
  ENTERPRISE 64,128         :
        PLOT Zahl2,Zahl3, ELLIPSE Zahl4,Zahl5
> MSX-STANDARD              :
        CIRCLE STEP (Zahl2,Zahl3),Zahl4,Zahl1,Zahl6,Zahl5,(Zahl4/Zahl5)
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

CLEAR

Befehlsart: Anweisung

CLEAR

Beschreibung: Dieser Befehl löscht alle Variablenspeicher. Die Zahlenvariablen erhalten den Wert Ø. Aus den String-Variablen werden leere oder "Null-Strings". Alle Daten werden getilgt.

Beispiel:

```
10 a=5
20 PRINT a
30 CLEAR
40 PRINT a
```

Als Lösung erhält man

```
5
Ø
```

Anmerkung: CLEAR kann nicht in einer FOR...TO - NEXT-Schleife verwendet werden. Das folgende Beispielprogramm (Schneider CPC 464) führt zur Fehlermeldung Unexpected NEXT in 4Ø.

```
10 FOR i=1 TO 20
20 PRINT 3*i
30 CLEAR
40 NEXT i
```

CLEAR löscht auch alle Definitionen, die mit DEF-Anweisungen (siehe DEF FN, siehe DEF SEG) vorgenommen wurden.

Anmerkung zur Kompatibilität: In einigen BASIC-Dialekten gibt es die Version

CLEARZahl

Dabei gibt die **Zahl** die Speicherplätze an, die für Zeichenketten reserviert werden.

Kompatibilitätsliste:

	ATARI 600,800,130XE	: CLR
	COMMODORE VC20,64	: CLR
	COMMODORE 16,116,PLUS4	: CLR
	COMMODORE 128	: CLR
›	ENTERPRISE 64,128	: CLEAR
›	MSX-STANDARD	: CLEAR
›	SCHNEIDER CPC464	: CLEAR
›	SCHNEIDER CPC664,CPC6128	: CLEAR
›	SINCLAIR ZX81	: CLEAR
›	SINCLAIR SPECTRUM	: CLEAR

CLG

Befehlsart: Anweisung

CLG **Zahl**

Beschreibung: Der Bildschirmhintergrund wird mit der Farbe eines "Farbstiftes" ausgefüllt. Die Nummer dieses "Farbstiftes" gibt man mit Hilfe der **Zahl** an. Sie ist vorher durch einen INK-Befehl (siehe INK) festgelegt worden. Schrift und Schrifthintergrund behalten dabei aber die Werte INK Ø (Schrifthintergrund) und INK 1 (Schrift).

Setzt man den Befehl ohne Angabe einer Farbstiftnummer (**Zahl**) ein, wird der Farbstift des letzten CLG-Befehls benutzt. Hat der Benutzer vorher allerdings noch keinen CLG-Befehl eingegeben, benutzt der Rechner automatisch den Farbstift Ø.

Beim Ausführen der CLG-Anweisung werden alle Zeichen auf dem Bildschirm gelöscht, wie beim CLS-Kommando (siehe CLS). Jedoch bleibt der Cursor in seiner alten Position stehen und wird nicht in das linke obere Eck gesetzt.

Beispiel:

```
10 CLS
20 INK 0,24:INK 1,0:INK 2,2
30 CLG 0
40 PRINT "Farbe 0"
50 INPUT a$:REM Verzoegerung; nur /enter/ druecken
60 CLG 1
70 PRINT "Farbe 1"
80 INPUT a$: REM Verzoegerung; nur /enter/ druecken
90 CLG 2
100 PRINT "Farbe 2"
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	CLG Zahl
› SCHNEIDER CPC664,CPC6128	:	CLG Zahl
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

CLOAD

Befehlsart: Kommando

CLOAD "**Programmname**"

Beschreibung: Ein Programm mit dem **Programmnamen** wird auf der Kassette gesucht und geladen. Das Programm muß zuvor mit CSAVE (siehe dort) abgespeichert worden sein. Der Befehl SAVE (siehe dort) benutzt ein anderes Format.

Beispiel:

CLOAD "Superprg"

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  CLOAD
  COMMODORE VC20,64         :  LOAD "Programmname"
  COMMODORE 16,116,PLUS4    :  LOAD "Programmname"
  COMMODORE 128             :  LOAD "Programmname"
  ENTERPRISE 64,128         :  LOAD "Programmname"
› MSX-STANDARD              :  CLOAD "Programmname"
  SCHNEIDER CPC464          :  LOAD "Programmname"
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :  LOAD "Programmname"
  SINCLAIR SPECTRUM         :  LOAD "Programmname"
```

CLOAD

Befehlsart: Kommando

CLOAD

Beschreibung: Ein auf der Kassette mit dem CSAVE- oder SAVE-Befehl (siehe CSAVE, siehe SAVE) abgespeichertes Programm wird geladen.

Ein im Speicher vorhandenes Programm wird durch das neue ersetzt, und alle noch vorhandenen Variablen werden gelöscht. Das Kommando kann das Format haben

CLOAD "**Dateiname**"

Beispiel:

CLOAD "Programm 1"

Die Kassette muß zuvor eingelegt und das Band an die richtige Stelle gebracht werden. Nachdem man die PLAY-Taste des Recorders und anschließend eine beliebige Taste der Tastatur gedrückt hat, wird das Programm geladen.

Das Kommando kann auch innerhalb eines Programmes verwendet werden.

Beispiel:

```
100 INPUT "WOLLEN WIR EIN SPIEL MACHEN";AN$
110 IF AN$="JA" THEN CLOAD "SPIEL 1"
```

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  CLOAD
  COMMODORE VC20,64         :  LOAD "Programmname"
  COMMODORE 16,116,PLUS4    :  LOAD "Programmname"
  COMMODORE 128             :  LOAD "Programmname"
  ENTERPRISE 64,128         :  LOAD "Programmname"
› MSX-STANDARD              :  CLOAD "Programmname"
  SCHNEIDER CPC464          :  LOAD "Programmname"
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :  LOAD "Programmname"
  SINCLAIR SPECTRUM         :  LOAD "Programmname"
```

CLOG

Befehlsart: Funktion

CLOG(**Zahl**)

Beschreibung: Mit CLOG berechnet man den Zehnerlogarithmus einer **Zahl**.

Die **Zahl** muß dabei größer als Ø sein, andernfalls kommt eine Fehlermeldung zurück. (Siehe auch LOG1Ø).

Beispiel:

```
PRINT CLOG(1Ø)
```

Anmerkung: Der Befehl sollte nur angewandt werden, wenn man mit den Exponentialfunktionen vertraut ist.

Kompatibilitätsliste:

› ATARI 600,800,130XE	(Anweisung)
COMMODORE VC20,64	
COMMODORE 16,116,PLUS4	
COMMODORE 128	
ENTERPRISE 64,128	(Anweisung)
MSX-STANDARD	
SCHNEIDER CPC464	
SCHNEIDER CPC664,CPC6128	
SINCLAIR ZX81	
SINCLAIR SPECTRUM	

CLOSE

Befehlsart: Anweisung

CLOSE **Dateinummer**

Beschreibung: Die Datei mit der **Dateinummer** wird verständigt und geschlossen. Sie wurde vorher mit dem OPEN-Befehl (siehe OPEN) eröffnet.

Beispiel:

CLOSE 2

schließt die Datei mit der Dateinummer 2.

Anmerkung: Die Ausführung von Kommandos wie NEW, RUN (siehe NEW, siehe RUN) usw. schließt automatisch alle noch offenen Dateien in einem Programm. Auch der END-Befehl (siehe END) schließt die Dateien, während sie von RUN STOP, ESCAPE, BREAK, STOP usw. nicht geschlossen werden.

Anmerkungen zur Kompatibilität: Die **Dateinummern** und die Anzahl der Dateien, die mit OPEN und CLOSE geöffnet bzw. geschlossen werden können, sind von Computertyp zu Computer

tertyp verschieden. Sie liegen zum Beispiel für den ZX Spectrum zwischen 4 und 15, beim VC 2Ø oder C64 zwischen 1 und 255.

In einigen BASIC-Dialekten (z. B. IBM) schließt die Anweisung CLOSE ohne Angabe einer **Dateinummer** alle offenen Dateien.

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  CLOSE #Dateinummer
> COMMODORE VC20,64         :  CLOSE Dateinummer
> COMMODORE 16,116,PLUS4    :  CLOSE Dateinummer
> COMMODORE 128             :  CLOSE Dateinummer
> ENTERPRISE 64,128         :  CLOSE #Dateinummer
> MSX-STANDARD              :  CLOSE #Dateinummer
  SCHNEIDER CPC464          :  CLOSEIN  bzw.  CLOSEOUT
  SCHNEIDER CPC664,CPC6128  :  CLOSEIN  bzw.  CLOSEOUT
  SINCLAIR ZX81             :
> SINCLAIR SPECTRUM         :  CLOSE #Dateinummer
```

CLOSEIN

Befehlsart: Anweisung

CLOSEIN

Beschreibung: Der Befehl schließt die Eingabedatei vom Diskettenlaufwerk oder Recorder, die vorher mit OPENIN"**Dateiname**" eröffnet wurden.

Mit OPENIN (siehe dort!) und CLOSEIN kann man beispielsweise Unterprogramme zum Einlesen von Dateien schreiben.

Beispiel:

```
890 OPENIN "ZUFALLSWORTE"
900 FOR X=1 TO 10
910 INPUT #9, WORT$
920 PRINT A$;" IST IM ";WORT$(X)
930 NEXT
940 CLOSEIN
950 RETURN
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :  CLOSE #Kanalnummer
  COMMODORE VC20,64         :  CLOSE Kanalnummer
  COMMODORE 16,116,PLUS4    :  CLOSE Kanalnummer
  COMMODORE 128             :  CLOSE Kanalnummer
  ENTERPRISE 64,128         :  CLOSE #Kanalnummer
  MSX-STANDARD              :  CLOSE #Kanalnummer
> SCHNEIDER CPC464          :  CLOSEIN
> SCHNEIDER CPC664,CPC6128  :  CLOSEIN
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :  CLOSE #Kanalnummer
```

CLOSEOUT

Befehlsart: Anweisung

CLOSEOUT

Beschreibung: Der Befehl schließt jede Ausgabedatei zum Diskettenlaufwerk oder Recorder.

Beispiel:

```
890 OPENOUT "REIMWORTE"
900 FOR X= 1 TO 10
910 WRITE #9, WORT1$(X),WORT2$(X)
920 CLOSEOUT
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :  CLOSE #Kanalnummer
  COMMODORE VC20,64         :  CLOSE Kanalnummer
  COMMODORE 16,116,PLUS4    :  CLOSE Kanalnummer
  COMMODORE 128             :  CLOSE Kanalnummer
  ENTERPRISE 64,128         :  CLOSE #Kanalnummer
  MSX-STANDARD              :  CLOSE #Kanalnummer
> SCHNEIDER CPC464          :  CLOSEOUT
> SCHNEIDER CPC664,CPC6128  :  CLOSEOUT
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :  CLOSE #Kanalnummer
```

CLR

Befehlsart: Anweisung

CLR

Beschreibung: Mit dieser Anweisung werden alle im Hauptspeicher befindlichen Daten gelöscht. Das Programm als solches bleibt dabei aber bestehen.

Beispiel:

```
10 A=23:B=12
20 PRINT A,B
30 CLR
40 PRINT A,B
RUN
 23           12
 0            0

READY.
```

Anmerkung: Durch Eingabe und Ausführung von RUN (siehe dort!) wird diese Anweisung automatisch durchgeführt.

Die Anweisung CLR macht auch alle Definitionen ungültig, die mit DEF FN (siehe dort!) vorgenommen worden sind.

Die Anweisung CLR läßt sich nicht innerhalb einer FOR... TO-NEXT-Schleife (siehe dort!) verwenden.

Anmerkung zur Kompatibilität: Die Anweisung CLR darf man nicht mit der Verwendung der CLR-Taste verwechseln. Diese Taste kann verschiedene Funktionen haben, zum Beispiel den Bildschirm löschen beim C64 oder Löschen eines Zeichens, auf dem der Cursor steht, beim Schneider 464/664.

Kompatibilitätsliste:

```
› ATARI 600,800,130XE        :  CLR
› COMMODORE VC20,64          :  CLR
› COMMODORE 16,116,PLUS4     :  CLR
› COMMODORE 128              :  CLR
  ENTERPRISE 64,128          :  CLEAR
  MSX-STANDARD               :  CLEAR
  SCHNEIDER CPC464           :  CLEAR
  SCHNEIDER CPC664,CPC6128   :  CLEAR
  SINCLAIR ZX81              :  CLEAR
  SINCLAIR SPECTRUM          :  CLEAR
```

CLS

Befehlsart: Kommando

CLS
CLS# Zahl

Beschreibung: CLS (Clearscreen) löscht den Bildschirm, indem alle seine Stellen mit der Farbe des Hintergrundes gefüllt werden. Der Cursor wird dabei (im Gegensatz zu CLG (siehe dort!) in die linke obere Ecke gesetzt.

Beispiel:

```
10 CLS
20 INPUT "1. zahl";a
30 INPUT "2. zahl";b
40 PRINT a;" * ";b;" = ";
50 INPUT c
60 IF c=a*b THEN PRINT "richtig!":GOSUB 100:GOTO 10
80 CLS: PRINT "falsch!":GOTO 40
100 FOR t=1 TO 700:NEXT:RETURN
```

Mit dem Format
CLS #Zahl
kann man beim Schneider CPC 464 bzw. 664 den anhand von #Zahl angegebenen Bildschirmbereich mit der für den Hin-

tergrund festgelegten Farbe löschen. Die **Zahl** bezeichnet dabei die Ein- und Ausgabeeinheit.

Beispiel:

```
10 CLS
20 A$ = "Der Anfuehrer einer Expedition":PRINT A$
30 INPUT "fertig (=j)";c$
40 IF c$="j" THEN CLS
50 INPUT B$
60 IF B$=A$ THEN PRINT "Richtig! Weiter!"
70 FOR t=1 TO 500: NEXT
80 CLS #1:GOTO 100
90 PRINT "Leider Falsch. Nochmal:":GOTO 20
100 C$ ="wurde vom archaeologischen"
110 PRINT C$
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	PRINT CHR$(147)
COMMODORE 16,116,PLUS4	:	PRINT CHR$(147)
COMMODORE 128	:	PRINT CHR$(147)
ENTERPRISE 64,128	:	PRINT CHR$(26)
› MSX-STANDARD	:	CLS
› SCHNEIDER CPC464	:	CLS #Zahl
› SCHNEIDER CPC664,CPC6128	:	CLS #Zahl
› SINCLAIR ZX81	:	CLS
› SINCLAIR SPECTRUM	:	CLS

CMD

Befehlsart: Kommando

CMD **Zahl,Parameterliste**

Beschreibung: Bildschirmausgaben, die mit PRINT oder LIST (siehe jeweils dort) ausgeführt werden, leitet man mit dem Befehl CMD auf ein Peripheriegerät um, zum Beispiel auf den Drucker.

Die **Zahl** gibt die Nummer jenes Datenkanals an, der durch den OPEN-Befehl (siehe dort) geöffnet wurde. Ihr Wert kann zwischen 1 und 255 liegen.

Mit Hilfe der **Parameterliste** kann man ähnlich wie beim PRINT-Befehl Texte, Zahlen oder andere Zeichenketten ausgeben. Die Liste darf gegebenenfalls auch fehlen.

Beispiel:

```
OPEN 1,4:CMD1:LIST:PRINT 1:CLOSE1
```

Diese Befehlssammlung bewirkt die Ausgabe eines Programmlistings auf den Drucker.

Statt CMD1 läßt sich auch schreiben

```
CMD1,"Programmlisting"
```

Man gibt damit dem Listing eine Überschrift.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
› COMMODORE VC20,64	:	CMD Zahl,Parameterliste
› COMMODORE 16,116,PLUS4	:	CMD Zahl,Parameterliste
› COMMODORE 128	:	CMD Zahl,Parameterliste
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

CODE

Befehlsart: Anweisung

CODE= String
CODE Variablenname= String

Beschreibung: Diese Anweisung wird in Verbindung mit Maschinen-Unterprogrammen verwendet. Wir können mit ihr einen **String** an die Adresse kopieren, die durch den Speicherplatz festgelegt ist. Dann läßt sich im Maschinenprogramm mit 4P diese Variable anhand der Adresse aufrufen. Wird mit dem **Variablennamen** noch eine Zahlenvariable angegeben, wird dieser Variablen die Adresse zugewiesen, die durch den Zeiger festgelegt ist.

Beispiel:

```
CODE = A$
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	CODE=String CODE Variable=String
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

CODE

Befehlsart: Funktion

CODE String
CODE "Textausdruck"

Beschreibung: Der Befehl entspricht der ASC-Funktion (siehe dort).
Für das erste Zeichen des **String**s wird die entsprechende ASC-Nummer zurückgegeben.

Beispiele:

```
I=CODE A$

PRINT CODE "A"
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :  ASC(String)
  COMMODORE VC20,64         :  ASC(String)
  COMMODORE 16,116,PLUS4    :  ASC(String)
  COMMODORE 128             :  ASC(String)
  ENTERPRISE 64,128         :
  MSX-STANDARD              :  ASC(String)
  SCHNEIDER CPC464          :  ASC(String)
  SCHNEIDER CPC664,CPC6128  :  ASC(String)
> SINCLAIR ZX81             :  CODE String
> SINCLAIR SPECTRUM         :  CODE String
```

COLLECT

Befehlsart: Kommando

COLLECT DZahl 1 ON UZahl 2
COLLECT DZahl 1,UZahl 2

Beschreibung: Mit diesem Kommando wird eine Diskette "gesäubert". Das heißt, sämtliche nicht korrekt geschlossenen Dateien werden geschlossen, und das Inhaltsverzeichnis wird auf den aktuellen Stand gebracht.

Zahl 1 gibt die Nummer des Laufwerkes wieder. Es sind also nur die beiden Werte Ø und 1 sinnvoll.

Zahl 2 wird nur benötigt, falls man mit zwei Einzellaufwerken arbeitet. Sie gibt die Geräteadresse an. Sinnvolle Werte sind 7, 8, 9 und 1Ø.

Bei Doppellaufwerken reicht es aus, den Befehl wie folgt zu verwenden:

COLLECT DZahl 1

Beispiel:

COLLECT DØ

Die Diskette in Laufwerk Ø wird gesäubert.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	COLLECT DZahl1 ON UZahl2
› COMMODORE 128	:	COLLECT DZahl1 ON UZahl2
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

COLLISION

Befehlsart: Anweisung

COLLISION **Zahl,Zeilennummer**

Beschreibung: Aufgrund einer Spritekollision wird eine Programmunterbrechung ein- oder abgeschaltet. Beim Einschalten wird eine Programmzeile mit der **Zeilennummer** angegeben, zu der das Programm im Fall einer Spritekollision verzweigt.

Der Befehl ist der Anweisung ON SPRITE GOSUB (siehe dort) verwandt.

Die **Zahl** gibt an, bei welchem Ereignis die Unterbrechung aktiviert oder inaktiviert wird. Es sind dabei die Zahlen 1, 2 und 3 erlaubt.

Es bedeutet

Zahl	
1	Sprite-Sprite-Kollision
2	Sprite-Zeichen-Kollision
3	Verzweigung bei Lightpen-Aktivierung

Die **Zeilennummer** muß nur angegeben werden, wenn man eine Unterbrechung aktivieren will. Sie hat einen ganzen Wert im Bereich von Ø bis 65535. Bei einer Spritekollision oder bei einer Lightpen-Aktivierung wird in das Unterprogramm gesprungen, welches in der Zeile mit der **Zeilennummer** beginnt.

Beispiele:

```
COLLISION 1,1ØØØØ

COLLISION 1
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  COLLISION Zahl,Zeilennummer
  ENTERPRISE 64,128         :
  MSX-STANDARD              :  ON SPRITE GOSUB Zeilennummer
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

COLOR

Befehlsart: Anweisung

COLOR **Zahl**

Beschreibung: In Abhängigkeit von der gewählten Grafik-Einstellung wird ein Farbregister ausgewählt, das für die nächste PLOT- oder DRAWTO-Anweisung (siehe jeweils dort) gilt. Damit lassen sich Punkte und Linien in unterschiedlichen Farben darstellen.

Beispiel:

```
10 GR 3
20 COLOR 2
30 PLOT 8,4:DRAWTO 28,10
```

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  COLOR Zahl
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

COLOR

Befehlsart: Anweisung

COLOR **Zahl 1,Zahl 2**

Beschreibung: Mit dem Befehl COLOR werden die Farben für den Hintergrund, den Vordergrund und den Rand festgelegt. Die **Zahl 1** bestimmt, ob der Hintergrund, der Vordergrund oder der Rand verändert werden soll, und die **Zahl 2** legt die Farbe nach der Farbnummer fest.

Beispiel:

COLOR Ø,4

In diesem Beispiel wird der Hintergrund grün.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	SETCOLOR Zahl1,Zahl2,6
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	COLOR Zahl1,Zahl2
› COMMODORE 128	:	COLOR Zahl1,Zahl2
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

COLOR

Befehlsart: Anweisung

COLOR **Zahl 1, Zahl 2, Zahl 3**

Beschreibung: Mit dieser Anweisung werden die Farben für den Vordergrund, den Hintergrund und den Rand festgelegt.
Die **Zahl 1** gibt die Farbe für den Vordergrund an, die **Zahl 2** den Hintergrund und die **Zahl 3** den Rand.

Beispiel:

```
COLOR 13,4,4
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	COLOR Zahl1,Zahl2,Zahl3
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

COM

Befehlsart: Anweisung

COM **Variablenname(Zahl 1,Zahl 2,Zahl 3...)**

Beschreibung: Mit dem COM-Befehl wird Speicherplatz für numerische oder alphanumerische Variablen reserviert.
Die Anweisung entspricht in allen Einzelheiten dem DIM-Befehl (siehe dort).

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	COM Variablenname(Zahl1,Zahl2,...)
		DIM Variablenname(Zahl1,Zahl2,...)
COMMODORE VC20,64	:	DIM Variablenname(Zahl1,Zahl2,...)
COMMODORE 16,116,PLUS4	:	DIM Variablenname(Zahl1,Zahl2,...)
COMMODORE 128	:	DIM Variablenname(Zahl1,Zahl2,...)
ENTERPRISE 64,128	:	DIM Variablenname(Zahl1,Zahl2,...)
MSX-STANDARD	:	DIM Variablenname(Zahl1,Zahl2,...)
SCHNEIDER CPC464	:	DIM Variablenname(Zahl1,Zahl2,...)
SCHNEIDER CPC664,CPC6128	:	DIM Variablenname(Zahl1,Zahl2,...)
SINCLAIR ZX81	:	DIM Variablenname(Zahl1,Zahl2,...)
SINCLAIR SPECTRUM	:	DIM Variablenname(Zahl1,Zahl2,...)

CONT

Befehlsart: Kommando

CONT

Beschreibung: Anhand des Kommandos CONT oder CONTINUE (siehe dort!) kann man ein mit STOP (siehe dort!), bzw. mit der ESC-, BREAK-, RUNSTOP- oder STOP-Taste unterbrochenes Programm fortsetzen. Es wird im Anschluß an die Stelle wieder aufgenommen, an der es unterbrochen wurde.

Um das Programm mit CONT fortsetzen zu können, darf man allerdings während der Unterbrechung keinen Eingriff in das Programm selbst vornehmen. Es dürfen also keine Zeilen gelöscht, hinzugefügt oder abgeändert werden.

Erlaubt ist hingegen während der Unterbrechung die Abfrage von Variablen und deren Veränderung. Man darf den Bildschirm löschen und das Programm auflisten.

Beispiel:

```
10 A=2
20 PRINT A + A;
30 A=A+1
40 GOTO 20
```

CONT

```
RUN
  4   6   8  10  12  14  16  18  20  22  24
 26  28  30  32  34  36  38  40  42  44
 46  48  50  52  54  56  58  60  62  64
 66  68
BREAK IN 20
READY.
CONT
 70  72  74  76  78  80  82  84  86  88
 90  92  94  96  98 100 102 104 106
108 110 112 114 116
BREAK IN 20
READY.
```

Anmerkung: Das Kommando CONT oder CONTINUE darf man nicht anwenden, falls das Programm mit einer Fehlermeldung abgebrochen wurde. Es erscheint auf dem Bildschirm als Fehlermeldung:

? CAN'T CONTINUE ERROR
cannot CONTinue
oder ähnliches.

Mit CONT läßt sich ein Programm, das zu Ende abgearbeitet worden ist, nicht neu starten.

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  CONT
› COMMODORE VC20,64         :  CONT
› COMMODORE 16,116,PLUS4    :  CONT
› COMMODORE 128             :  CONT
  ENTERPRISE 64,128         :  CONTINUE
› MSX-STANDARD              :  CONT
› SCHNEIDER CPC464          :  CONT
› SCHNEIDER CPC664,CPC6128  :  CONT
› SINCLAIR ZX81             :  CONT
› SINCLAIR SPECTRUM         :  CONT
```

CONTINUE

Befehlsart: Kommando

CONTINUE

Beschreibung: Mit dem Kommando CONTINUE kann man ein mit STOP, bzw. mit der ESC-, BREAK-, RUNSTOP- oder STOP-Taste unterbrochenes Programm nach der Unterbrechungsstelle wieder fortsetzen. Es wird dann bis zum Ende bzw. bis zu einer neuerlichen Unterbrechung weiter abgearbeitet (siehe CONT).

Beispiel:

```
1Ø FOR i=1 TO 2Ø
2Ø PRINT "***i= ";i
3Ø STOP
4Ø NEXT
CONTINUE
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	CONT
COMMODORE VC20,64	:	CONT
COMMODORE 16,116,PLUS4	:	CONT
COMMODORE 128	:	CONT
› ENTERPRISE 64,128	:	CONTINUE
MSX-STANDARD	:	CONT
SCHNEIDER CPC464	:	CONT
SCHNEIDER CPC664,CPC6128	:	CONT
SINCLAIR ZX81	:	CONT
SINCLAIR SPECTRUM	:	CONT

COPY

Befehlsart: Kommando

COPY

Beschreibung: Der Bildschirminhalt wird über einen angeschlossenen Drucker ausgegeben.

Beispiel:

```
11Ø INPUT "ZUFRIEDEN";A$
12Ø IF A$="JA" THEN COPY
```

Anmerkung zur Kompatibilität: Dieses COPY-Kommando ist nicht zu verwechseln mit COPY-Befehlen anderer BASIC-Dialekte (siehe dort).

Kompatibilitätsliste:

ATARI 600,800,130XE	:
COMMODORE VC20,64	:
COMMODORE 16,116,PLUS4	:
COMMODORE 128	:
ENTERPRISE 64,128	:
MSX-STANDARD	:
SCHNEIDER CPC464	:
SCHNEIDER CPC664,CPC6128	:
SINCLAIR ZX81	: COPY
SINCLAIR SPECTRUM	:

COPY

Befehlsart: Kommando

COPY DZahl 1,"Programmname 1" TO DZahl 2,"Programmname 2" ON UZahl 3

COPY DZahl 1,"Programmname" TO DZahl 2,"Programmname",
UZahl 3

Beschreibung: Eine Datei oder mehrere Dateien werden kopiert. Das geschieht entweder innerhalb einer Diskette oder von einer Quelldiskette auf eine Zieldiskette. Die Zieldiskette muß dabei bereits formatiert sein (im Gegensatz zu BACKUP; siehe dort!).

Zahl 1 bezeichnet die Laufwerksnummer der Quelldiskette. Es sind die Werte Ø und 1 möglich.

Programmname 1 ist der Name der Datei, die kopiert werden soll.

Zahl 2 bezeichnet die Laufwerksnummer der Zieldiskette. es sind die Werte Ø und 1 möglich.

Programmname 2 ist der neue Name der kopierten Datei. Hat man ihn nicht angegeben, so wird die Datei unter ihrem ursprünglichen Namen kopiert.

Zahl 3 ist die Geräteadresse bei Verwendung von zwei Einzellaufwerken.

Beispiel:

COPY "TEST" TO "VERSUCH"
kopiert die Datei TEST auf dieselbe Diskette unter dem Namen VERSUCH

COPY DØ TO D1
kopiert alle Dateien der Diskette in Laufwerk Ø auf die Diskette in Laufwerk 1

COPY DØ, "TEST" TO D1,"VERSUCH"
kopiert die Datei TEST von der Diskette in Laufwerk Ø auf die Diskette in Laufwerk 1 unter dem Namen VERSUCH

COPY DØ,"TEST" TO D1
kopiert die Datei TEST von der Diskette in Laufwerk Ø auf die Diskette in Laufwerk 1

COPY DØ, "T*" TO D1
kopiert alle Dateien der Diskette in Laufwerk Ø, deren

Namen mit dem Buchstaben T beginnen, auf die Diskette in Laufwerk 1

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :
    COPY DZahl1,"Programmname1" TO DZahl2,"Programmname2",UZahl3
> COMMODORE 128             :
    COPY DZahl1,"Programmname1" TO DZahl2,"Programmname2",UZahl3
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

COS

Befehlsart: Funktion

COS(**Zahl**)

Beschreibung: Das Ergebnis der mathematischen Funktion ist die Wiedergabe der Winkelfunktion COSINUS.

Die **Zahl** ist dabei der Winkel, der im Bogenmaß angegeben wird. Das Ergebnis ist ein Wert zwischen -1 und 1.

Beispiel:

```
PRINT COS (Ø)          A=2 :  PRINT COS(A)
 1                      - .4161468
```

Anmerkung: Die Funktion sollte nur verwendet werden, wenn man mit den Winkelfunktionen vertraut ist.

Um das Ergebnis in Grad wiedergeben zu können, muß man es mit 57.29578 multiplizieren.

Anmerkung zur Kompatibilität: Bei ZX81, ZX Spectrum und Sinclair QL hat der Befehl das Format COS **Zahl**.

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  COS(Zahl)
› COMMODORE VC20,64         :  COS(Zahl)
› COMMODORE 16,116,PLUS4    :  COS(Zahl)
› COMMODORE 128             :  COS(Zahl)
› ENTERPRISE 64,128         :  COS(Zahl)
› MSX-STANDARD              :  COS(Zahl)
› SCHNEIDER CPC464          :  COS(Zahl)
› SCHNEIDER CPC664,CPC6128  :  COS(Zahl)
› SINCLAIR ZX81             :  COS Zahl
› SINCLAIR SPECTRUM         :  COS Zahl
```

COSH

Befehlsart: Funktion

COSH(**Zahl**)

Beschreibung: Die Funktion gibt den COSINUSHYPERBOLICUS einer **Zahl** wieder.

Beispiele:

```
PRINT COSH(1)

A=COSH(B)
```

Anmerkung: Die Funktion sollte man nur verwenden, wenn man mit den Hyperbelfunktionen vertraut ist.

Anmerkung zur Kompatibilität: Bei Rechnern, die über diesen Befehl nicht verfügen, kann er wie folgt simuliert werden:

COSH(Zahl)

entspricht

(EXP(Zahl)+EXP(-Zahl))/2

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	COSH(Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

COT

Befehlsart: Funktion

COT(**Zahl**)

Beschreibung: Die Funktion gibt den Cotangens der **Zahl** zurück. Im normalen Fall geschieht dies im Bogenmaß, bei einigen Rechnern allerdings auch im Winkelmaß. Die **Zahl** darf dabei nicht Ø oder ein Vielfaches von 18Ø Grad bzw. PI (π) sein.

Beispiele:

```
PRINT COT(1)

I=COT(A)
```

Anmerkung zur Kompatibilität: Bei Rechnern, die über diesen Befehl nicht verfügen, kann der Cotangens wie folgt errechnet werden:

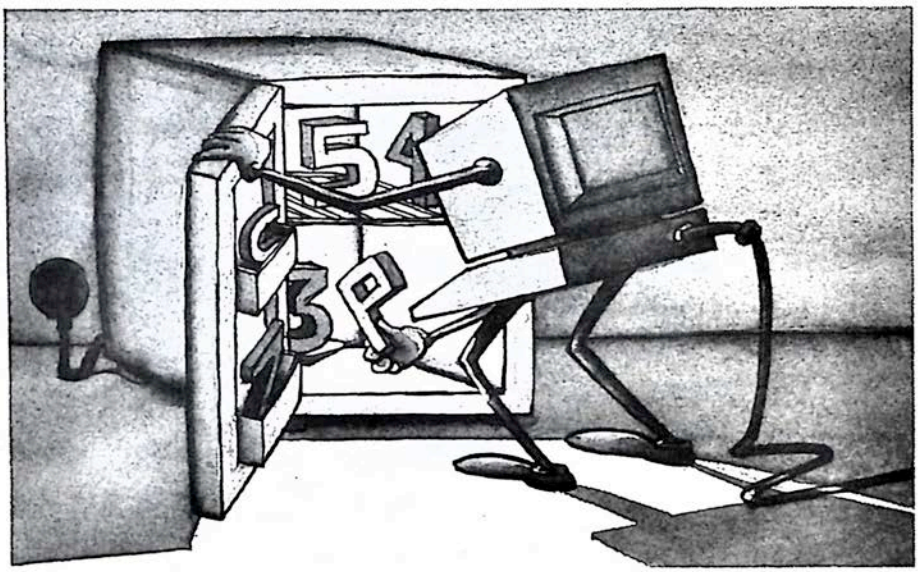
COT(**Zahl**)

entspricht

1/TAN(**Zahl**)

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	COT(Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	



Manchmal treibt ein plötzlich auftretender Heißhunger meinen Computer nachts noch an den Kühlschrank...

Stadt- und Landesbibliothek Dortmund

CREAL

Befehlsart: Funktion

CREAL(**Zahl**)

Beschreibung: Dies ist der "Umkehrbefehl" zu CINT (siehe dort!). Beim CINT-Befehl wird eine Zahl mit Fließkomma in einen ganzzahligen Wert gerundet. CREAL dagegen gibt den "reellen" Wert einer Fließkommazahl wieder, kann jedoch die durch CINT erfolgte Rundung nicht rückgängig machen.

Für die Programmierpraxis hat der Befehl keine Bedeutung.

Beispiel:

```
10 DEFINT N:N=4.087
20 PRINT N
30 PRINT CREAL(N)
40 DEFREAL N:N=4.087
50 PRINT N
60 PRINT CREAL(N)
```

```
 4
 4
 4.087
 4.087
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEADER CPC464          :  CREAL(Zahl)
> SCHNEIDER CPC664,CPC6128  :  CREAL(Zahl)
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

CSAVE

Befehlsart: Kommando

CSAVE
CSAVE "Programmname"

Beschreibung: Ein im Rechnerspeicher befindliches Programm wird auf der Kassette im Recorder abgespeichert. Die Kassette muß zuvor eingelegt und das Band an die richtige Stelle gebracht werden. Nachdem man die PLAY- und RECORD-Tasten und anschließend eine beliebige Taste auf der Tastatur des Rechners gedrückt hat, wird das Programm abgespeichert.

Mit CLOAD (siehe dort) kann man das Programm wieder laden. CSAVE kann auch innerhalb eines Programmes verwendet werden.

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  CSAVE
  COMMODORE VC20,64         :  SAVE "Programmname"
  COMMODORE 16,116,PLUS4    :  SAVE "Programmname"
  COMMODORE 128             :  SAVE "Programmname"
  ENTERPRISE 64,128         :  SAVE "Programmname"
› MSX-STANDARD              :  CSAVE "Programmname",Zahl
  SCHNEIDER CPC464          :  SPEED WRITE Zahl:SAVE "Programmname"
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :  SAVE "Programmname"
  SINCLAIR SPECTRUM         :  SAVE "Programmname"
```

CSAVE

Befehlsart: Kommando

CSAVE "**Programmname**",**Zahl**

Beschreibung: Mit dem CSAVE-Kommando wird ein Programm unter dem **Programmnamen** auf Kassette abgespeichert.
Mit der **Zahl** kann man zwischen der Übertragungsgeschwindigkeit von 12ØØ BAUD (**Zahl** = 1) und 24ØØ BAUD (**Zahl** = 2) wählen. BAUD gibt die Übertragungsgeschwindigkeit in Bit pro Sekunde an.

Beispiel:

CSAVE "Programm1",2

Anmerkung zur Kompatibilität: Bei einigen Rechnern (siehe Kompatibilitätsliste) ist die Wahl der Übertragungsgeschwindigkeit nicht möglich. Wird kein **Programmname** angegeben, verhalten sich die einzelnen Rechner unterschiedlich.

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  CSAVE
  COMMODORE VC20,64         :  SAVE "Programmname"
  COMMODORE 16,116,PLUS4    :  SAVE "Programmname"
  COMMODORE 128             :  SAVE "Programmname"
  ENTERPRISE 64,128         :  SAVE "Programmname"
› MSX-STANDARD              :  CSAVE "Programmname",Zahl
  SCHNEIDER CPC464          :  SPEED WRITE Zahl:SAVE "Programmname"
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :  SAVE "Programmname"
  SINCLAIR SPECTRUM         :  SAVE "Programmname"
```

CSC

Befehlsart: Funktion

CSC(**Zahl**)

Beschreibung: Die Funktion gibt den Cosekans der **Zahl** zurück. Die **Zahl** als Argument des Cosekans darf nicht die Werte (2k+1)*9Ø Grad beziehungsweise (2k+1)*PI/2 annehmen (k entspricht dabei irgendeiner ganzen Zahl).

Beispiel:

```
PRINT CSC(1)

Y=CSC(X)
```

Anmerkung: Diese Funktion sollte nur verwendet werden, wenn man mit den Winkelfunktionen vertraut ist.

Anmerkung zur Kompatibilität: Die Funktion läßt sich, falls der entsprechende Befehl nicht zur Verfügung steht, folgendermaßen simulieren:

CSC(**Zahl**)

entspricht

1/COS(**Zahl**)

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	CSC(Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

CSNG

Befehlsart: Funktion

CSNG(**Zahl**)

Beschreibung: Eine **Zahl** wird in eine Zahl von einfacher Genauigkeit (6 Stellen) umgewandelt.

Die **Zahl** kann ein fester Wert wie .234565644 sein. Sie kann aber auch durch eine definierte Variable (wie zum Beispiel A) oder einen Rechenausdruck vertreten werden (z. B. 5/3).

Beispiel:

```
PRINT CSNG(5/3)
 1.66666
READY
```

Bei

```
10 ...
20 a! = CSNG(b#)
30 ...
```

entspricht b# der Zahl mit doppelter Genauigkeit; a! entspricht der Zahl mit einfacher Genauigkeit.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	CSGN(Zahl)
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

CSRLIN

Befehlsart: Funktion

CSRLIN

Beschreibung: Zurückgegeben wird die Nummer der Zeile, in der sich augenblicklich der Cursor befindet.

Man erhält also die senkrechte Position (=Zeilennummer) des Cursors. Mit dem Wert läßt sich der Cursor neu positionieren, falls erforderlich. Die oberste Zeile des Bildschirms hat dabei die Nummer Ø. Jede weitere Zeile wird um 1 erhöht.

Beispiel:

```
CLS:PRINT CSRLIN
 Ø
OK
```

Anmerkung zur Kompatibilität: Der Befehl VPOS (siehe VPOS; siehe auch Kompatibilitätsliste) erzielt dasselbe Ergebnis. Beim C64 ist er durch einen PEEK-Befehl (PRINT PEEK (214) zu ersetzen).

Kompatibilitätsliste:

ATARI 600,800,130XE	
COMMODORE 64	PEEK(214)
COMMODORE 16,116,PLUS4	
COMMODORE 128	
ENTERPRISE 64,128	
MSX-STANDARD	CSRLIN
SCHNEIDER CPC464	VPOS(#0)
SCHNEIDER CPC664,CPC6128	VPOS(#0)
SINCLAIR ZX81	
SINCLAIR SPECTRUM	

DATA

Befehlsart: Anweisung

DATA **Textausdruck 1,Zahl 1....**
DATA **Zahl 1,Zahl 2,Textausdruck 1....**

Beschreibung: Auf die Anweisung DATA folgt eine Reihe von Datenelementen, die durch die READ-Anweisung (siehe READ) abgerufen werden. Es wird ein internes Datenfeld geschaffen.

Es kann sich dabei sowohl um Zahlenwerte als auch um Textausdrücke handeln, wobei man Textausdrücke (meist) nicht in Anführungszeichen setzen muß. Ausnahmen: Der Textausdruck enthält ein Leerzeichen, einen Doppelpunkt oder ein Komma, was vom Programm als Instruktion mißverstanden würde.

Ist zwischen zwei Kommas kein Wert bzw. kein Textausdruck angegeben, so wird dies entweder als Ø gewertet (numerische Auswertung) oder als ein Leerstring (alpha-numerische Auswertung - also Strings).

Beispiel:

```
10 READ AN,BN,A$,B$,C$
20 PRINT AN,BN:PRINT A$,B$,C$
30 DATA 13,12,DIESES,BUCH,"GEHOERT:"

RUN
 13           12
DIESES       BUCH        GEHOERT:

READY.
```

Anmerkung: Um die Programmausführung zu beschleunigen, fügt man die DATA-Anweisungen möglichst am Ende des Programmes an.

Die DATA-Anweisung wird während der Ausführung des Programmes nicht beachtet, solange keine READ-Anweisung erreicht wurde.

Anmerkungen zur Kompatibilität: Unterschiedlich ist vor allem die Anzahl der Datenelemente, die von einer DATA-Anweisung im jeweiligen BASIC-Dialekt entgegengenommen wird. Einige Computer akzeptieren nur Datenelemente, die nicht über eine Programmzeile hinausgehen.

Eine Besonderheit des Sinclair SPECTRUM: Mit

SAVE"**Dateiname**" DATA a(**Zahl**)

und

LOAD"**Dateiname**" DATA a (**Zahl**)

können Datenfelder gespeichert bzw. geladen werden.

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  siehe oben und COM
› COMMODORE VC20,64         :  siehe oben
› COMMODORE 16,116,PLUS4    :  siehe oben
› COMMODORE 128             :  siehe oben
› ENTERPRISE 64,128         :  siehe oben
› MSX-STANDARD              :  siehe oben
› SCHNEIDER CPC464          :  siehe oben
› SCHNEIDER CPC664,CPC6128  :  siehe oben
  SINCLAIR ZX81             :
› SINCLAIR SPECTRUM         :  siehe oben
```

DATE

Befehlsart: Anweisung

DATE **String**

Beschreibung: Mit diesem Befehl geben wir dem Rechner das aktuelle Tagesdatum vor. Bei Tageswechsel springt der Rechner um einen Tag weiter.

Der **String** enthält das Datum in Form von 8 Zeichen. Er hat das Format "JJJJMMTT", wobei die mit "J" bezeichneten Stellen für die Angabe des Jahres, die mit "M" bezeichneten für die Angabe des Monats und die mit "T" bezeichneten für die Angabe des Tages reserviert sind. Der 6. Dezember 1986 wäre folglich angegeben als

19861Ø06.

Aus diesem Beispiel ist ersichtlich, daß man die nicht benötigten Stellen bei Angabe von Monat und Tag mit einer Ø auffüllen muß.

Beispiel:

```
10 DATE "19800228"
20 TIME "23:59:00"
30 PRINT DATE$,TIME$
40 GOTO 30
```

Anmerkung: Der Rechner erkennt selbständig bei der Eingabe eines Datums Schaltjahre.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	DATE String
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

DATE$

Befehlsart: Funktion

DATE$

Beschreibung: Mit Hilfe dieser Funktion läßt sich das aktuelle Datum abfragen, vorausgesetzt, daß man es mit der DATE-Anweisung (siehe dort) eingegeben hat.

Man kann sich das Datum mit dem PRINT-Befehl (siehe dort) rausgeben lassen oder dieses Datum einer Stringvariablen zuweisen.

Hintereinander und ohne Trennung werden gedruckt: vierstellige Jahreszahl, zweistellige Angabe des Monats und zweistellige Angabe des Tages.

Beispiele:

```
PRINT DATE$
 19851126
READY
```

Das Beispiel gibt das Datum des 26. November 1985 wieder.

```
A$=DATE$
```

```
100 CLS
110 PRINT "JAHR: ";LEFT$(DATE$,4)
120 PRINT "MONAT: ";MID$(DATE$,5,2)
130 PRINT "TAG: ";RIGHT$(DATE$,2)
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	DATE$
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

DCLEAR

Befehlsart: Anweisung

DCLEAR D**Zahl 1** ON U**Zahl 2**

Beschreibung: Mit dem DCLEAR-Befehl werden alle offenen Kanäle des Diskettenlaufwerks geschlossen, nicht jedoch die offenen Dateien.

Die **Zahl 1** gibt die Laufwerksnummer an. Man muß sie nur bei Verwendung mehrerer Laufwerke angeben.

Die **Zahl 2** gibt die Geräteadresse an. Sie ist nur bei Verwendung mehrerer Einzellaufwerke erforderlich. Normalerweise liegt sie zwischen 7 und 1Ø, kann aber auch Werte zwischen 4 und 15 annehmen.

Beispiele:

```
DCLEAR
DCLEAR DØ
DCLEAR D1 ON U9
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  DCLEAR DZahl1 ON UZahl2
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

DEC

Befehlsart: Funktion

DEC(**String**)

Beschreibung: Der Befehl DEC wandelt eine hexadezimale Zahl, die als **String** vorliegt, in eine dezimale Zahl um.
Dabei darf der **String** maximal 4 Zeichen umfassen und nur Zeichen enthalten, die zum hexadezimalen Zahlensystem gehören. Der **String** darf also nur aus folgenden Zeichen bestehen: Ø, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

Beispiel:

```
PRINT DEC("FFFF")
 65535

READY
```

Anmerkung: Die Zeichen des **String**s müssen ohne Verwendung der SHIFT-Taste eingegeben werden. Wenn man sich also im Kleinschriftmodus befindet, dürfen im **String** keine großen Buchstaben vorkommen.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  DEC(String)
> COMMODORE 128             :  DEC(String)
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

DEF

Befehlsart: Anweisung

DEF **Funktionsname (Parameterliste)**

Beschreibung: Der Befehl ähnelt der Anweisung DEF FN (siehe dort). Er bietet allerdings mehr Möglichkeiten.

1. Anwendung
Der Befehl wird wie DEF FN verwendet.

2. Anwendung
Eine Funktion wird innerhalb eines DEF-Blocks definiert, wie anhand des Beispiels erläutert.

Beispiel:

```
10 DEF RUNDUNG(ZAHL,STELLE)
20 ZAHL=ZAHL*10^STELLE
30 ZAHL=INT(ZAHL+0.5)
40 ZAHL=ZAHL/10^STELLE
50 END DEF
60 INPUT X : REM GEBE AN DIESER STELLE EINE ZAHL EIN
70 INPUT Y : REM GEBE AN DIESER STELLE DIE GEWUENSCHTE STELLENZAHL AN
80 CALL RUNDUNG(X,Y)
90 PRINT "GERUNDETE ZAHL=";X
```

In Zeile 10 wird der Name der Funktion mit der Parameterliste angegeben.
Die Zeilen 20 bis 40 geben die Operation innerhalb der Funktion an.
Die Zeile 50 bedingt das Funktionsende.
Die Zeile 60 verlangt die Eingabe der Zahl, die gerundet werden soll.
Die Zeile 70 verlangt die Eingabe der Stellen-Anzahl.
Zeile 80 ruft die definierte Funktion auf.
Zeile 90 gibt die gerundete Zahl aus.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	siehe oben
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

DEFDBL

Befehlsart: Anweisung

DEFDBL **Variablenbereich**

Beschreibung: Alle Variablen, die mit den Buchstaben beginnen, die im **Variablenbereich** niedergelegt sind, werden als Zahlen mit doppelter Genauigkeit (= 14 Stellen) wiedergegeben.

Im **Variablenbereich** darf man nur Einzelbuchstaben verwenden. Dabei unterscheiden wir nicht zwischen Groß- und Kleinschreibung. Die Variable a ist im **Variablenbereich** gleichbedeutend mit der Variablen A.

Mögliche Verwendungen des Befehls:

DEFDBL a	Alle Variablen, die mit a (oder A) beginnen,
DEFDBL a,b,f	Alle Variablen, die mit a,b,f (oder A,B,F) beginnen,
DEFDBL a-d	Alle Variablen, die mit a,b,c,d (oder A,B,C,D) beginnen,
DEFDBL a-d,j	Alle Variablen, die mit a,b,c,d,j (oder A,B,C,D,J) beginnen,
	sind Variablen mit doppelter Genauigkeit.

Beispiel:

```
10 DEFBDL a-d
20 a= 2.41234568:a$=momentaufnahme"
30 PRINT a,b,a$
```

Anmerkung: Fügt man ein sogenanntes Typendeklarationszeichen an den Variablennamen (bei MSX %,$,!,#), wird die durch DEFDBL definierte Anweisung ignoriert.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
> MSX-STANDARD	:	DEFDBL Variablenbereich
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

DEF FN

Befehlsart: Anweisung

DEF FN **Funktionsname**(Parameterliste)=**Funktion**

Mit DEF FN kann man eigene Funktionen definieren. Sie lassen sich später im Programm mit dem Befehl FN (siehe dort!) rufen.

Die Funktion wird mit dem **Funktionsnamen** bezeichnet. In der Parameterliste werden die in der **Funktion** benötigten Variablen übergeben.

Beispiel:

```
10 DEF FN RUNDUNG(ZAHL,STELLE)= INT(ZAHL*10^STELLE+0.5)/10^STELLE
20 INPUT "GIB MIR BITTE EINE ZAHL MIT KOMMASTEL-  LEN EIN. (BEIM COMPUTERN VERWE
NDET MAN  DAFUER EINEN PUNKT.)";X
30 INPUT "AUF WIEVIEL STELLEN NACH DEM KOMMA       (PUNKT) SOLL GERUNDET WERDEN "
;Y
40 Y=INT(Y) : REM MACHT AUS Y EINE GANZE ZAHL
50 X=FN RUNDUNG (X,Y)
60 PRINT "DIE GERUNDETE ZAHL LAUTET: ";X
70 END
```

Anmerkung: CLR oder NEW löschen auch die durch DEF FN definierte Funktion.

Die Definition der Funktion sollte im Programm nur einmal durchlaufen werden.

Eine Funktion innerhalb der Definition einer Funktion zu verwenden, ist nicht möglich.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE      :
> COMMODORE VC20,64        :  DEF FNFunktionsname(Parameterliste)=Funktion
> COMMODORE 16,116,PLUS4   :  DEF FNFunktionsname(Parameterliste)=Funktion
> COMMODORE 128            :  DEF FNFunktionsname(Parameterliste)=Funktion
  ENTERPRISE 64,128        :  DEF Funktionsname(Parameterliste)=Funktion
> MSX-STANDARD             :  DEF FNFunktionsname(Parameterliste)=Funktion
> SCHNEIDER CPC464         :  DEF FNFunktionsname(Parameterliste)=Funktion
> SCHNEIDER CPC664,CPC6128 :  DEF FNFunktionsname(Parameterliste)=Funktion
  SINCLAIR ZX81            :
> SINCLAIR SPECTRUM        :  DEF FN Funktionsname(Parameterliste)=Funktion
```

DEFINT

Befehlsart: Anweisung

DEFINT **Variablenbereich**

Beschreibung: Alle Variablen, die mit den im **Variablenbereich** niedergelegten Buchstaben beginnen, werden als ganzzahlige Werte wiedergegeben. Fließkommazahlen werden auf- bzw. abgerundet.

Im **Variablenbereich** darf man nur Einzelbuchstaben verwenden. Dabei wird nicht nach Groß- oder Kleinschreibung unterschieden; die Variable a ist im **Variablenbereich** folglich gleichbedeutend mit der Variablen A.

Mögliche Verwendungen des Befehls:

DEFINT a	Alle Variablen, die mit a (oder A) beginnen
DEFINT a,b,f	Alle Variablen, die mit a,b,f (oder A,B,F) beginnen,

DEFINT

DEFINT a-d — Alle Variablen, die mit a,b,c,d (oder A,B,C,D) beginnen,

DEFINT a-d,j — Alle Variablen, die mit a,b,c,d,j (oder A,B,C,D,J) beginnen,

sind Integervariablen.

Beispiel:

```
10 DEFINT A-D
20 A=2.4:B=3.78
30 PRINT A;B
RUN
 2  4
```

Anmerkung: Im allgemeinen legt man die Definition der Integervariablen an den Anfang des Programmes.

Nachdem mit DEFINT der Variablentyp festgelegt wurde, muß man ihn, wie bei den anderen Definitionsanweisungen (siehe DEFSNG, siehe DEFSTR, siehe DEFDBL) im Programm nicht mehr besonders ausweisen.

Kompatibilitätsliste:

```
' ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  DEFINT Variablenbereich
> SCHNEIDER CPC464          :  DEFINT Variablenbereich
> SCHNEIDER CPC664,CPC6128  :  DEFINT Variablenbereich
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

DEFREAL

Befehlsart: Kommando

DEFREAL **Variablenbereich**

Beschreibung: Alle Variablen, die mit den Buchstaben beginnen, die im **Variablenbereich** niedergelegt sind, werden als reelle Zahlen wiedergegeben. Es sind also die aus der Mathematik bekannten "Fließkommazahlen".

Im **Variablenbereich** darf man nur Einzelbuchstaben verwenden. Dabei unterscheiden wir nicht zwischen Groß- oder Kleinschreibung. Die Variable a ist im **Variablenbereich** gleichbedeutend mit der Variablen A.

Mögliche Verwendung des Befehls:

DEFREAL a	Alle Variablen, die mit a (oder A) beginnen,
DEFREAL a,b,f	Alle Variablen, die mit a,b,f (oder A,B,F) beginnen,
DEFREAL a-d	Alle Variablen, die mit a,b,c,d (oder A,B,C,D) beginnen,
DEFREAL a-d,j	Alle Variablen, die mit a,b,c,d,j (oder A,B,C,D,J) beginnen,

sind Variablen von reellem Wert.

Dieser Befehl erscheint auf den ersten Blick überflüssig, da der Computer ohnehin alle Variablen, die nicht anders definiert sind, als reelle Zahlen behandelt. Der Befehl wird jedoch notwendig, sobald man einen **Variablenbereich** mit DEFINT (siehe dort) als Bereich von Integervariablen definiert hat. Um nun wieder mit reellen, also Fließkomma-Zahlen arbeiten zu können, müssen die Variablen mit DEFREAL neu definiert werden.

Beispiel:

```
10 DEFINT a
20 a=2.5
30 b=10/a
40 PRINT b
50 DEFREAL a
60 a=2.5
70 b=10/a
80 PRINT b
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :  DEFREAL Variablenbereich
> SCHNEIDER CPC664,CPC6128  :  DEFREAL Variablenbereich
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

DEFSNG

Befehlsart: Anweisung

DEFSNG **Variablenbereich**

Beschreibung: Alle Variablen, die mit den Buchstaben beginnen, die im **Variablenbereich** niedergelegt sind, werden als Zahlen mit einfacher Genauigkeit (= 6 Stellen) wiedergegeben.

Im **Variablenbereich** darf man nur Einzelbuchstaben verwenden. Dabei unterscheiden wir nicht zwischen Groß- oder Kleinschreibung. Die Variable a ist im **Variablenbereich** gleichbedeutend mit der Variablen A.

Mögliche Verwendung des Befehls:

DEFSNG a — Alle Variablen, die mit a (oder A) beginnen,

DEFSNG a,b,f Alle Variablen, die mit a,b,f (oder A,B,F) beginnen,
DEFSNG a-d Alle Variablen, die mit a,b,c,d (oder A,B,C,D) beginnen,
DEFSNG a-d,j Alle Variablen, die mit a,b,c,d,j (oder A,B,C,D,J) beginnen,

sind Variablen mit einfacher Genauigkeit.

Beispiel:

```
10 DEFSNG a-d
20 a=2.41234568:b=5/3:a$="max"
30 PRINT a,b,a$
```

Anmerkung: Fügt man ein sogenanntes Typendeklarationszeichen an den Variablennamen (bei MSX %,$,!,#), wird die durch DEFSNG definierte Anweisung ignoriert.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
> MSX-STANDARD	:	DEFSNG Variablenbereich
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

DEFSTR

Befehlsart: Anweisung

DEFSTR **Variablenbereich**

Beschreibung: Alle Variablen, die mit den Buchstaben beginnen, die im **Variablenbereich** angegeben sind, werden als

Stringvariablen behandelt. Dabei wird das normalerweise geforderte $-Zeichen nach dem Namen nicht mehr unbedingt benötigt. Will man verhindern, daß beim Programmieren Verwechslungen von Zahlen- und Stringvariablen vorkommen, darf man es durchaus angeben. Auf keinen Fall aber darf man jedoch die bei Stringvariablen erforderlichen Anführungszeichen vergessen. Sonst erfolgt entweder eine Fehlermeldung, oder der Programmierfehler macht sich erst später beim Ablauf eines Programms unliebsam bemerkbar.

Im **Variablenbereich** dürfen nur Einzelbuchstaben verwendet werden. Dabei unterscheiden wir nicht zwischen Groß- und Kleinschreibung. Die Variable a ist im **Variablenbereich** gleichbedeutend mit der Variablen A.

Mögliche Verwendung des Befehls:

DEFSTR a	Alle Variablen, die mit a (oder A) beginnen,
DEFSTR a,b,f	Alle Variablen, die mit a,b,f (oder A,B,F) beginnen,
DEFSTR a-d	Alle Variablen, die mit a,b,c,d (oder A,B,C,D) beginnen,
DEFSTR a-d,j	Alle Variablen, die mit a,b,c,d,j (oder A,B,C,D,J) beginnen,
	werden als Stringvariablen behandelt.

Beispiel:

```
10 DEFSTR a,b
20 a="Peter"
30 b$="Georg"
40 PRINT a,b
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  DEFSTR Variablenbereich
> SCHNEIDER CPC464          :  DEFSTR Variablenbereich
> SCHNEIDER CPC664,CPC6128  :  DEFSTR Variablenbereich
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

DEFUSR

Befehlsart: Anweisung

DEFUSR**Zahl 1**=**Zahl 2**

Beschreibung: Für das Maschinensprache-Unterprogramm **Zahl 1** wird als Startadresse die **Zahl 2** definiert.

Die **Zahl 1** legt das Unterprogramm fest, dessen Startadresse neu definiert wird. Ihr Wert kann zwischen Ø und 9 liegen. Ruft man das Unterprogramm später mit USR (siehe dort!) auf, gibt die **Zahl 1** die Nummer des Unterprogrammes an. Der Rechner greift dann auf jene Startadresse zurück, die durch DEFUSR definiert worden ist.

Die **Zahl 2** für die Startadresse des Maschinensprache-Unterprogrammes muß einen Wert haben, der zwischen Ø und 65535 liegt.

Beispiel:

DEFUSR1=2ØØØØ

Das Unterprogramm 1 erhält die Startadresse 2ØØØØ.

Anmerkung: Diesen Befehl sollte man verwenden, wenn man über Erfahrung mit der Maschinensprache verfügt. Fehler bei der Definition können zum Absturz des Rechners führen, falls das Maschinenprogramm mit USR aufgerufen wird.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	DEFUSR Zahl1-Zahl2
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

DEG

Befehlsart: Anweisung

DEG

Beschreibung: Durch diesen Befehl wird statt des Bogenmaßes bei den Winkelfunktionen (siehe SIN, COS, TAN, ACOS, ACS, ASIN, ASN, ATN) das Winkelgradmaß verwendet. Normalerweise rechnet der Computer im Bogenmaß. Zu lernen sind die Winkelfunktionen jedoch zumeist anhand des Winkelmaßes. Der Befehl ist daher nützlich, sofern man sich mit den Winkelfunktionen vertraut machen will.

Beispiel:

```
10 RAD : REM(---RAD)
20 PRINT "IM BOGENMASS LAUTET DER WERT FUER ATN(1)";ATN(1)
30 DEG
40 PRINT "IM WINKELGRADMASS LAUTET DER WERT FUER ATN(1)";ATN(1)
IM BOGENMASS LAUTET DER WERT FUER ATN(1)
 0.785398163
IM WINKELGRADMASS LAUTET DER WERT FUER ATN(1)  45
```

Anmerkung zur Kompatibilität: Rechner, die nicht über diesen Befehl verfügen, können das Ergebnis mit folgenden Gleichungen erzielen:

1. Bogenmaß(B) in Winkelgradmaß(W) W=B*180/3.14159265
2. Winkelgradmaß(W) in Bogenmaß(B) B=W*3.14159265/180

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  DEG
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :  OPTION ANGLE DEGREES
  MSX-STANDARD              :
> SCHNEIDER CPC464          :  DEG
> SCHNEIDER CPC664,CPC6128  :  DEG
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

DEG

Befehlsart: Funktion

DEG(**Zahl**)

Beschreibung: Mit der Funktion DEG wird die **Zahl** vom Bogenmaß ins Winkelgradmaß umgewandelt.

Anmerkung: Diese Funktion darf man nicht mit der Anweisung DEG (siehe dort) verwechseln, die eine generelle Umschaltung vom Bogenmaß auf das Winkelgradmaß vornimmt.

Anmerkung zur Kompatibilität: Bei Rechnern, deren BASIC nicht über diesen Befehl verfügt, kann er wie folgt simuliert werden:

```
DEG(Zahl)
```

entspricht

```
Zahl*18Ø/PI
```

Beispiele:

```
PRINT DEG(1)

A=DEG(Ø.5)
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  DEG(Zahl)
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

DELETE

Befehlsart: Kommando

DELETE **Zeilennummer 1-Zeilennummer 2**

Beschreibung: Es werden alle Zeilen zwischen **Zeilennummer 1** und **Zeilennummer 2** einschließlich gelöscht. Man kann sie dann nur wieder neu eingeben.

Beispiel:

DELETE 1ØØ -15Ø	löscht alle Programmzeilen zwischen 1ØØ und 15Ø
DELETE 8Ø	löscht die Programmzeile 8Ø
DELETE	löscht das gesamte Programm
DELETE 8Ø-	löscht das gesamte Programm ab Zeile 8Ø
DELETE -12Ø	löscht das gesamte Programm bis einschließlich Zeile 12Ø

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	DELETE Zeilennummer1-Zeilennummer
› COMMODORE 128	:	DELETE Zeilennummer1-Zeilennummer
› ENTERPRISE 64,128	:	DELETE Zeilennummer1 TO Zeilennummer
› MSX-STANDARD	:	DELETE Zeilennummer1-Zeilennummer
› SCHNEIDER CPC464	:	DELETE Zeilennummer1-Zeilennummer
› SCHNEIDER CPC664,CPC6128	:	DELETE Zeilennummer1-Zeilennummer
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

DI EI

Befehlsart: Anweisung

DI EI

Beschreibung: Mit dem Befehl EVERY (siehe dort!) lassen sich in regelmäßigen Zeitabständen Unterprogramme aufrufen. Dabei ist es allerdings möglich, daß ein Programmteil nicht vollständig durchlaufen, sondern durch den EVERY-Befehl unterbrochen wird. Dies kann unerwünschte Folgen für die Programmabarbeitung haben, da zum Beispiel der Bildschirmaufbau zerstört wird.

Um derartige Probleme zu vermeiden, setzt man den Befehl DI (Disable Interrupt) ein. Setzt man den Befehl an den Anfang einer Befehlsreihe, so kann diese durch den EVERY-Befehl nicht mehr unterbrochen werden, und zwar so lange, bis das Programm den Befehl EI (Enable Interrupt) erreicht.

Der EI-Befehl hebt die Wirkung des DI-Befehls auf. Hat man vorher keinen DI-Befehl gegeben, so zeigt EI keine Wirkung.

Beispiel:

```
10 EVERY 10 GOSUB 500
20 DI
30 FOR I=0 TO 1000
40 PRINT I
50 NEXT I
60 EI
70 FOR I=0 TO 2000
80 PRINT I
90 NEXT I
100 END
500 REM ---BILDSCHIRM LOESCHEN---
510 CLS
520 RETURN
```

Anmerkung: Der DI-Befehl unterdrückt auch alle anderen Unterbrechungen, ausgenommen Programmfehler und die entspre-

chenden Fehlermeldungen sowie die Befehle STOP (siehe dort!), END (siehe dort!) und Drücken der ESC-Taste.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :  INTERVAL OFF
> SCHNEIDER CPC464          :  DI
> SCHNEIDER CPC664,CPC6128  :  DI
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

DIM

Befehlsart: Anweisung

DIM **Variablenname(Zahl 1,Zahl 2,Zahl 3...)**

Beschreibung: Bei einer größeren Anzahl von gleichartigen Variablen arbeitet man statt einer langen Reihe einfacher Zuweisungen mit indizierten Variablen. Indizierte Variablen bestehen aus einem **Variablennamen** und mehreren positiven ganzen Zahlen oder einer positiven ganzen Zahl. Sie werden in Klammern angefügt und durch Kommata getrennt. Statt einem Wert

```
A$
```

sind nun als indizierte Variablen möglich

```
A$(1)
A$(2)
A$(3)
.
.
A$(x)
```

Mit dem DIM-Befehl reservieren wir den für die indizierte Variable benötigten Speicherplatz. **Zahl 1** drückt aus, welche Größe x bei A$(x) annehmen kann. Mit dem Befehl

```
DIM A$(5)
```

hat man ein eindimensionales Feld mit fünf Variablen-Möglichkeiten definiert. Bei den meisten BASIC-Dialekten müssen allerdings Felder von mehr als 1Ø Elementen erst vorher DIMensioniert werden.

Es lassen sich sowohl String-Variablen als auch numerische Variablen in Form indizierter Variablen ausdrücken. Mit numerischen, indizierten Variablen läßt sich ebenso rechnen wie mit einfachen Variablen.

Beispiel:

```
10 DIM a(3,4)
20 FOR i=1 TO 3
30 FOR j=1 TO 4
40 INPUT a(i,j)
50 PRINT a(i,j)*2
60 NEXT j
70 NEXT i
```

Noch mehr Variablen der gleichen Art sind in zweidimensionalen Feldern zu verarbeiten. Ein Feld, das mit

```
DIM A(5,5)
```

dimensioniert wurde, hat als Variablen-Möglichkeiten:

A(1,1), A(1,2), A(1,3), A(1,4), A(1,5),
A(2,1), A(2,2), A(2,3), A(2,4), A(2,5),
A(3,1), A(3,2), A(3,3), A(3,4), A(3,5),
A(4,1), A(4,2), A(4,3), A(4,4), A(4,5),
A(5,1), A(5,2), A(5,3), A(5,4), A(5,5).

In manchen BASIC-Dialekten kommt dabei noch die Ø als Indizierungswert hinzu. Daher gilt für die Anzahl der reservierten Speicherplätze bei einem zweidimensionalen Feld folgende Formel:

DIM A(x,y) reserviert (x+1) * (y+1) Speicherplätze.

Entsprechend vielfältig sind die Variablen-Möglichkeiten bei drei-, vier- oder x-dimensionalen Feldern.

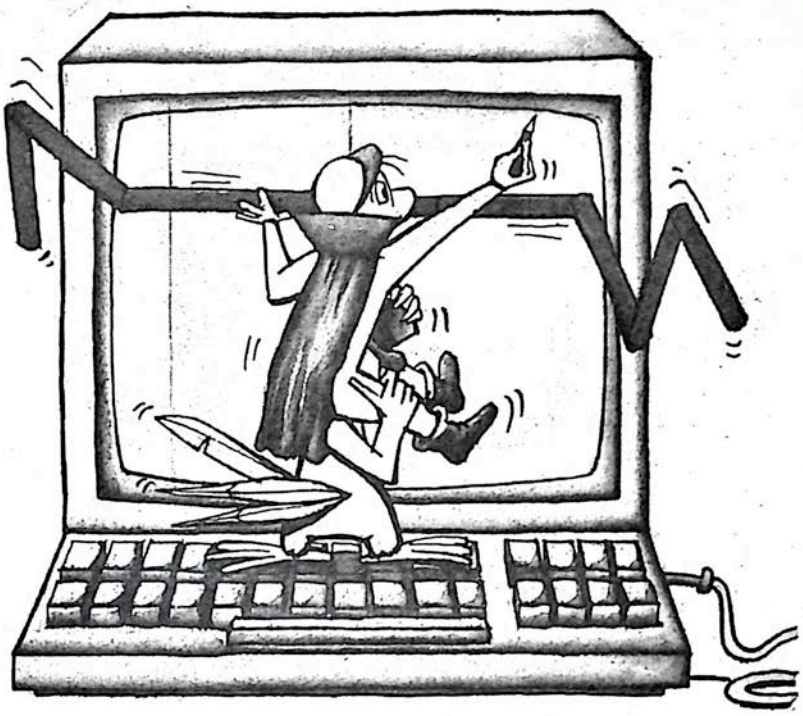
Der Programmiervorteil bei der Verwendung von indizierten Variablen kommt daher, daß die Indices nicht nur Zahlenwerte, sondern auch Variable sein können, die man innerhalb von Schleifen verändern oder auch durch Zufallsgenerator definieren kann.

Beispiel:

```
10 REM miniadressendatei
20 DIM a$(3,5)
30 FOR i=1 TO 5
40 INPUT "name";a$(1,i)
50 INPUT "strasse";a$(2,i)
60 INPUT "wohnort";a$(3,i)
70 NEXT i
80 PRINT "nr   name,strasse,wohnort"
90 FOR i=1 TO 5
100 PRINT i;
110 FOR j=1 TO 3
120 PRINT a$(j,i)
130 NEXT j
140 PRINT
150 NEXT i
160 END
```

Kompatibilitätsliste:

```
› ATARI 600,800,130XE      :  COM Variablenname(Zahl1,Zahl2,...)
                              DIM Variablenname(Zahl1,Zahl2,...)
› COMMODORE VC20,64        :  DIM Variablenname(Zahl1,Zahl2,...)
› COMMODORE 16,116,PLUS4   :  DIM Variablenname(Zahl1,Zahl2,...)
› COMMODORE 128            :  DIM Variablenname(Zahl1,Zahl2,...)
› ENTERPRISE 64,128        :  DIM Variablenname(Zahl1,Zahl2,...)
› MSX-STANDARD             :  DIM Variablenname(Zahl1,Zahl2,...)
› SCHNEIDER CPC464         :  DIM Variablenname(Zahl1,Zahl2,...)
› SCHNEIDER CPC664,CPC6128 :  DIM Variablenname(Zahl1,Zahl2,...)
› SINCLAIR ZX81            :  DIM Variablenname(Zahl1,Zahl2,...)
› SINCLAIR SPECTRUM        :  DIM Variablenname(Zahl1,Zahl2,...)
```



DIRECTORY

Befehlsart: Kommando

DIRECTORY D**Zahl 1** ON U**Zahl 2**,"**Programmname**"
DIRECTORY D**Zahl 1**,U**Zahl 2**,"**Programmname**"

Beschreibung: Nach Eingabe des Kommandos wird das Inhaltsverzeichnis der Diskette auf dem Bildschirm abgebildet.

Im Gegensatz zu LOAD"$",8 (siehe LOAD) wird dabei das im Speicher befindliche Programm nicht gelöscht.

Die **Zahl 1** gibt die Laufwerknummer der Diskette an, deren Inhaltsverzeichnis gelistet werden soll.

Die **Zahl** 2 gibt die Geräteadresse des Laufwerks der Diskette an, deren Inhaltsverzeichnis gelistet werden soll.

Der **Programmname** gibt den Namen des Programmes an, das auf der Diskette gesucht werden soll. Fehlt er, werden alle auf der Diskette befindlichen Programme auf dem Bildschirm angezeigt. Gibt man an Stelle des **Programmnamens** nur einen Buchstaben oder mehrere Buchstaben und ein Sternchen ein, so werden alle Programme aufgelistet, deren Programmnamen mit diesem/diesen Buchstaben beginnen.

Beispiel:

```
DIRECTORY
```

zeigt auf dem Bildschirm alle Programme einer Diskette.

```
DIRECTORY DØ, "P*"
```

zeigt alle Programme auf dem Bildschirm, deren Dateiname mit dem Buchstaben P beginnen und die sich auf der Diskette in Laufwerk Ø befinden.

Anmerkung: Soll das Inhaltsverzeichnis ausgedruckt werden, muß man statt des Kommandos

```
  DIRECTORY
```

mit

```
  LOAD"$",8
```

das Directory in den Speicher laden. Es läßt sich mit

```
  OPEN1,4,Ø:CMD1:LIST:PRINT#1
```

auf den Drucker bringen.

Kompatibilitätsliste:

ATARI 600,800,130XE :	
COMMODORE VC20,64 :	
› COMMODORE 16,116,PLUS4 :	DIRECTORY DZahl1,UZahl2,"Programmname"
› COMMODORE 128 :	CATALOG DZahl1,UZahl2,"Programmname"
	DIRECTORY DZahl1,UZahl2,"Programmname"
ENTERPRISE 64,128 :	
MSX-STANDARD :	
SCHNEIDER CPC464 :	
SCHNEIDER CPC664,CPC6128 :	CAT
SINCLAIR ZX81 :	
SINCLAIR SPECTRUM :	CAT Zahl2

DISPLAY

Befehlsart: Anweisung

DISPLAY **Zahl Ø** AT **Zahl 1** FROM **Zahl 2** TO **Zahl 3**
DISPLAY GRAPHICS
DISPLAY TEXT

Beschreibung: Mit diesem Befehl kann der Benutzer zwischen den einzelnen Grafik-Betriebsarten hin und her schalten, ohne daß die Bildschirm-Bilder gelöscht werden. Der Befehl wirkt gleichsam wie ein Zettel, den man auf eine Information legt, den man aber auch wieder wegnehmen kann.

DISPLAY **Zahl Ø** AT **Zahl 1** FROM **Zahl 2** TO **Zahl 3**

blendet einen Teil des Textbildschirmes in den Grafikbildschirm ein. **Zahl Ø** entspricht der Kanalnummer (sie beträgt normalerweise 1Ø2). **Zahl 1** gibt die Zeile an, in der die erste eingeblendete Zeile steht. **Zahl 2** gibt die erste Zeile des Textbildschirmes an, dessen Zeilen eingeblendet werden, die **Zahl 3** die letzte.

DISPLAY GRAPHICS

blendet diese Information bei einem Grafikschirm wieder aus.

DISPLAY TEXT

blendet diese Information bei Textbetrieb wieder aus.

Beispiel:

DISPLAY 1Ø2: AT 1 FROM 1 TO 5

Dieser Befehl blendet die ersten fünf Zeilen des ursprünglichen Textbildschirmes ein.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	siehe Beschreibung
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

DLOAD

Befehlsart: Kommando

DLOAD "**Programmname**",D**Zahl 1** ON U**Zahl 2**
DLOAD "**Programmname**",D**Zahl 1**,U**Zahl 2**

Beschreibung: Ein Programm mit dem **Programmnamen** wird von der Diskette in den Speicher des Rechners geladen.

Mit diesem Befehl kann man auch innerhalb eines BASIC-Programmes ein Programm nachladen, das automatisch gestartet wird. Die alten Variablen bleiben erhalten und lassen sich somit weiterverarbeiten. Das neue Programm muß allerdings kürzer sein als das im Rechner befindliche Programm.

Die **Zahl 1** bezeichnet die Laufwerksnummer. Es sind nur die Zahlen Ø und 1 sinnvoll.

Die **Zahl 2** bezeichnet die Geräteadresse des Diskettenlaufwerkes. Sie muß angegeben werden, wenn man mehrere Einzellaufwerke verwendet.

Beispiele:

```
DLOAD "Versuch"
```

Es wird das Programm VERSUCH geladen.

```
DLOAD "T*"
```

Es wird das Programm geladen, dessen Name mit dem Buchstaben T beginnt.

```
A$="VERSUCH":DLOAD(A$)
```

Es wird das Programm VERSUCH geladen. Ist A$ ein Leerstring (A$=""), wird eine Fehlermeldung zurückgegeben.

```
DLOAD "Apfel",DØ
```

Es wird von der im Laufwerk Ø befindlichen Diskette das Programm APFEL geladen.

Anmerkung: Kassettenprogramme werden mit dem LOAD-Befehl (siehe LOAD) in den Rechner geladen.

Anmerkung zur Kompatibilität: Beim C64 wird von der Diskette mit dem Befehl

LOAD"**Programmname**",**Zahl 2**

geladen. **Zahl 2** gibt die Laufwerkadresse an. Normalerweise hat sie den Wert 8; bei einem zweiten Laufwerk sind aber auch die Werte 9 oder 1Ø möglich.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :  LOAD "D:Programmname"
  COMMODORE VC20,64         :  LOAD "Programmname",Zahl2
> COMMODORE 16,116,PLUS4    :  DLOAD "Programmname",DZahl1,UZahl2
> COMMODORE 128             :  DLOAD "Programmname",DZahl1,UZahl2
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :  LOAD "Programmname"
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :  LOAD "M";Zahl2;"Programmname"
```

DO...LOOP

Befehlsart: Anweisung

DO...LOOP
DO UNTIL **Ausdruck**...LOOP
DO...LOOP UNTIL **Ausdruck**
DO WHILE **Ausdruck**...LOOP
DO...LOOP WHILE **Ausdruck**

Beschreibung: Mit diesem Befehl wird eine Schleife so lange ausgeführt, bis aufgrund von definierten Bedingungen, etwa im **Ausdruck**, die Schleife verlassen wird. Diese Art der Schleife wurde der Programmiersprache PASCAL entlehnt.

Die Schleife beginnt mit dem Befehl DO und wird mit LOOP abgeschlossen. Je nach der Formulierung der Bedingung wird die Schleife ab DO fortgeführt.

Mit dem Zusatz UNTIL wird die Schleife so lange ausgeführt, bis die Bedingung für UNTIL erfüllt ist ("until" = engl. bis). Die Bedingung wird durch eine Vergleichsoperation gebildet (,=,), (siehe auch unter IF...THEN).

Mit dem Zusatz WHILE wird so lange in der Schleife verweilt, bis die Bedingung nicht mehr erfüllt ist.

Mit EXIT (siehe dort) kann man die Schleife vorzeitig verlassen.

Beispiel:

```
10 DO
20 PRINT "BIN ICH EIN KLUGER KOPF?"
30 INPUT AS
40 LOOP UNTIL AS="JA"
50 PRINT "GIB BITTE EINE ZAHL EIN"
60 INPUT I
70 DO WHILE X<I
80 PRINT X
90 X=X+1
100 LOOP
110 END
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  siehe Beschreibung
> COMMODORE 128             :  siehe Beschreibung
> ENTERPRISE 64,128         :  siehe Beschreibung
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

DOS

Befehlsart: Kommando

DOS

Beschreibung: Dieses Kommando ist nur bei Verwendung einer Diskettenstation sinnvoll.

Der Befehl DOS zeigt auf dem Bildschirm das DOS-Hauptmenü. Das DOS (Disc Operating System) organisiert die Verwaltung der Floppy. Im Hauptmenü werden die dazu benötigten Befehle aufgelistet.

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  DOS
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

DRAW

Befehlsart: Anweisung

DRAW **Zahl 1,Zahl 2,Zahl 3**

Beschreibung: Es wird eine Linie auf dem Bildschirm gezeichnet. Die aktuelle Position des Graphikcursors ist dabei der Startpunkt der Linie, deren Endpunkt durch **Zahl 1** (x-Koordinate, also Zeile) und **Zahl 2** (y-Koordinate, also Spalte) definiert ist. Liegt einer der Punkte außerhalb des Bildschirmbereiches, so wird die Linie bis an den Rand des Bildschirms gezeichnet; der Rest wird nicht beachtet.

Zahl 3 bezeichnet die Nummer des Farbstiftes. Man muß sie nicht unbedingt angeben. Durch den INK-Befehl (siehe INK) lassen sich neue Farben definieren und so mehrfarbige Graphiken erstellen.

Beispiel:

```
10 MODE 2
20 DRAW INT(RND*640),INT(RND*400)
30 FOR i=0 TO 100:NEXT i
40 GOTO 20
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :  DRAWTO Zahl1,Zahl2
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :  DRAW Zahl3, TO Zahl1,Zahl2
  COMMODORE 128             :  DRAW Zahl3, TO Zahl1,Zahl2
  ENTERPRISE 64,128         :  SET BEAM ON:PLOT Zahl1,Zahl2
  MSX-STANDARD              :  DRAW "MZahl1,Zahl"
› SCHNEIDER CPC464          :  DRAWR Zahl1,Zahl2,Zahl3
› SCHNEIDER CPC664,CPC6128  :  DRAWR Zahl1,Zahl2,Zahl3
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :  DRAW Zahl1,Zahl2
```

DRAWR

Befehlsart: Anweisung

DRAWR **Zahl 1,Zahl 2,Zahl 3**

Beschreibung: Es wird eine Linie auf dem Bildschirm gezeichnet. Die aktuelle Position des Graphicursors ist dabei der Startpunkt der Linie, die im durch **Zahl 1** und **Zahl 2** definierten Punkt endet. Dabei versetzt **Zahl 1** die x-Koordinate der ersten Cursor-Position um **Zahl 1** Zeilen und **Zahl 2** die y-Koordinate um **Zahl 2** Spalten.

Es entspricht somit die

x-Koordinate (neu) der **Zahl 1**+x-Koordinate Cursor (alt)
y-Koordinate (neu) der **Zahl 2**+y-Koordinate Cursor (alt)

Liegt einer der Punkte außerhalb des Bildschirmbereiches, so wird die Linie bis an den Rand des Bildschirmes gezeichnet; der Rest wird nicht zur Kenntnis genommen.

Zahl 3 bezeichnet die Nummer des Farbstiftes. Man muß sie nicht unbedingt angeben. Durch den INK-Befehl (siehe INK) lassen sich neue Farben definieren und so mehrfarbige Graphiken erstellen.

Beispiel:

```
10 xl=10:yl=10
20 MODE 2
30 DRAWR x,y
40 x=x+xl:IF x>=630 OR x<=0 THEN xl=-1*xl
50 DRAWR x,y
60 y=y+yl:IF y>=390 OR y<=0 THEN yl=-1*yl
70 GOTO 30
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	DRAWR Zahl1,Zahl2,Zahl3
› SCHNEIDER CPC664,CPC6128	:	DRAWR Zahl1,Zahl2,Zahl3
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

DRAWTO

Befehlsart: Anweisung

DRAWTO **Zahl 1,Zahl 2**

Beschreibung: Mit dem DRAWTO-Befehl wird eine Linie von der letzten Cursor-Stellung bis zu der von **Zahl 1** (x-Achse) und **Zahl 2** (y-Achse) bestimmten Stelle gezogen. Die x-Koordinate entspricht der Spalte, die y-Koordinate der Zeile.

Beispiel:

```
10 GRAPHICS 3
20 COLOR 1
30 PLOT 5,1
40 DRAWTO 34,16
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	DRAWTO Zahl1,Zahl2
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	DRAW Zahl3, TO Zahl1,Zahl2
COMMODORE 128	:	DRAW Zahl3, TO Zahl1,Zahl2
ENTERPRISE 64,128	:	SET BEAM ON:PLOT Zahl1,Zahl2
MSX-STANDARD	:	DRAW "MZahl1,Zahl"
SCHNEIDER CPC464	:	DRAWR Zahl1,Zahl2,Zahl3
SCHNEIDER CPC664,CPC6128	:	DRAWR Zahl1,Zahl2,Zahl3
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	DRAW Zahl1,Zahl2

DS

Befehlsart: Variable

DS

Beschreibung: Die Variable enthält die Fehlernummer für einen eventuell auftretenden Fehler beim Betrieb mit der Diskettenstation.

Beispiele:

```
I=DS

PRINT DS
```

Anmerkung: Der Text für die durch DS wiedergegebene Fehlernummer ist in der Variablen DS$ (siehe dort) abgelegt.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	ST
› COMMODORE 16,116,PLUS4	:	DS
› COMMODORE 128	:	DS
ENTERPRISE 64,128	:	EXTYPE
MSX-STANDARD	:	ERR
SCHNEIDER CPC464	:	ERR
SCHNEIDER CPC664,CPC6128	:	ERR
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

DS$

Befehlsart: Variable

DS$

Beschreibung: Die Variable enthält als String einen Fehlermeldungstext, der einen Fehler beim Betrieb mit der Diskettenstation beschreibt.

Beispiele:

```
PRINT DS$

A$=DS$
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  DS$
> COMMODORE 128             :  DS$
  ENTERPRISE 64,128         :  EXSTRING$(EXTYPE)
  MSX-STANDARD              :  ERR
  SCHNEIDER CPC464          :  ERR
  SCHNEIDER CPC664,CPC6128  :  ERR
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

DSAVE

Befehlsart: Kommando

```
DSAVE "Programmname",DZahl 1 ON UZahl 2
DSAVE "Programmname",DZahl 1,UZahl 2
```

Beschreibung: Ein Programm im Rechnerspeicher wird auf Diskette unter dem **Programmnamen** abgespeichert.

Statt des **Programmnamens** in Anführungszeichen kann man auch eine String-Variable in Klammern angeben.

Die **Zahl 1** bezeichnet die Laufwerksnummer. Es sind nur die Zahlen Ø und 1 sinnvoll.

Die **Zahl 2** bezeichnet die Geräteadresse des Diskettenlaufwerkes. Sie ist nur dann anzugeben, wenn mehrere Einzellaufwerke verwendet werden.

Beispiel:

```
DSAVE "VERSUCH"
```

Das Programm wird unter dem Namen VERSUCH abgespeichert.

```
A$="VERSUCH":DSAVE(A$)
```

Das Programm wird unter dem Namen VERSUCH abgespeichert.

```
DSAVE "APFEL",DØ
```

Es wird das Programm unter dem Namen APFEL auf der in Laufwerk Ø befindlichen Diskette abgespeichert.

Anmerkung: Kassettenprogramme können mit dem SAVE-Befehl (siehe SAVE) abgespeichert werden.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE        :  SAVE "D:Programmname"
  COMMODORE VC20,64          :  SAVE "Programmname",Zahl2
> COMMODORE 16,116,PLUS4     :  DSAVE "Programmname",DZahl1,UZahl2
> COMMODORE 128              :  DSAVE "Programmname",DZahl1,UZahl2
  ENTERPRISE 64,128          :
  MSX-STANDARD               :
  SCHNEIDER CPC464           :
  SCHNEIDER CPC664,CPC6128   :  SAVE "Programmname"
  SINCLAIR ZX81              :
  SINCLAIR SPECTRUM          :  SAVE "M";Zahl2;"Programmname"
```

EDIT

Befehlsart: Kommando

EDIT **Zeilennummer**

Beschreibung: Um eine bestimmte Zeile eines Programmes zu bearbeiten, bringen wir sie mit EDIT **Zeilennummer** auf den Bildschirm und geben sie zur Bearbeitung frei. Mit Cursorsteuerung und CLR-Taste lassen sich Fehler tilgen und berichtigen, indem man zusätzliche Zeichen und Befehle über die Tastatur eingibt.

Beispiel:

```
EDIT 40
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :  EDIT Zahl
> SCHNEIDER CPC664,CPC6128  :  EDIT Zahl
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

EI

Befehlsart: Anweisung

EI

Beschreibung: EI (Enable Interrupt) hebt die Wirkung des DI-Befehles (siehe DI; Disable Interrupt) auf, der die Unterbrechung eines Programmteiles durch den EVERY-Befehl (siehe EVERY) unmöglich macht.

Wurde vorher kein DI-Befehl gegeben, so zeigt EI keine Wirkung.

Beispiel:

```
10 EVERY 20 GOSUB 500
20 DI
30 MODE 2
40 FOR x=1 TO 34
50 FOR y=1 TO 34
60 DRAW x,y
70 NEXT y
80 NEXT x
90 EI
100 FOR x=1 TO 12
110 DRAW x,y
115 FOR t=1 TO 500:NEXT
120 NEXT x
130 END
500 REM ///bildschirm loeschen///
510 CLG
520 RETURN
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :  INTERVAL ON
> SCHNEIDER CPC464          :  EI
> SCHNEIDER CPC664,CPC6128  :  EI
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

EL

Befehlsart: Variable

EL

Beschreibung: Die Variable enthält die Zeilennummer, in der der BASIC-Interpreter zuletzt einen Fehler festgestellt hat (EL = Error in Line).

Beispiel:

```
I=EL

PRINT EL
```

Anmerkung: EL entspricht der Variablen ERL.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  EL
> COMMODORE 128             :  EL
  ENTERPRISE 64,128         :  EXLINE
  MSX-STANDARD              :  ERL
  SCHNEIDER CPC464          :  ERL
  SCHNEIDER CPC664,CPC6128  :  ERL
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

ELSE

Befehlsart: Anweisung

IF...THEN...ELSE

Beschreibung: Bei einer bedingten Verzweigung mit IF... THEN wird ELSE als Entscheidungsanweisung für den Fall verwendet, daß die Bedingung nicht erfüllt ist.
Das Format

IF **logischer Ausdruck** THEN **Anweisungen 1** ELSE **Anweisungen 2**

ergibt folgende Arbeitsweise:

Ist der **logische Ausdruck** mit "wahr" zu beantworten, dann werden die **Anweisungen 1** ausgeführt;
ist der logische Ausdruck mit "nicht wahr" zu beantworten, dann werden die **Anweisungen 2** ausgeführt.

Beispiel:

```
100 INPUT A$
110 INPUT B$
120 IF A$>B$ THEN PRINT A$ ELSE PRINT B$
```

Anmerkung zur Kompatibilität: Bei BASIC-Dialekten, die nicht über den ELSE-Befehl verfügen, kann diese Anweisung durch eine Kombination mit GOTO (siehe dort) ersetzt werden.
Für das obige Beispielprogramm sieht die Ausführung dann so aus:

```
100 INPUT A$
110 INPUT B$
120 IF A$<B$ THEN PRINT A$:GOTO 140
130 PRINT B$
140 PRINT"*"
```

```
END

RUN
? WERT
? FELD
FELD
*
READY.
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	IF ... THEN ... ELSE ...
COMMODORE 128	:	IF ... THEN ... ELSE ...
ENTERPRISE 64,128	:	IF ... THEN ... ELSE ...
MSX-STANDARD	:	IF ... THEN ... ELSE ...
SCHNEIDER CPC464	:	IF ... THEN ... ELSE ...
SCHNEIDER CPC664,CPC6128	:	IF ... THEN ... ELSE ...
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

END

Befehlsart: Kommando

END

Beschreibung: Das Programm wird beendet. Alle Dateien werden geschlossen. Im Gegensatz zu STOP (siehe dort!) wird keine BREAK-Meldung gezeigt. Der Rechner schaltet in den Direktmodus zurück.

Das END-Kommando muß nicht am virtuellen Ende des Programmes erscheinen.

Beispiel:

```
10 PRINT "WOLLEN WIR BEGINNEN? JA/NEIN"
20 INPUT A$
30 IF A$="JA" THEN GOTO 50
40 END
50 PRINT"ALSO LOS!"
```

Anmerkung: Das BASIC-Programm endet auch ohne END-Kommando mit dem Abarbeiten der letzten Programmzeile, falls in dieser kein Sprungbefehl erscheint. Mit CONT (siehe CONT) oder CONTINUE (siehe CONTINUE) lassen sich Programme, die an einen END-Befehl gelangt sind, wieder starten. Das Programm beginnt dann wieder mit der Abarbeitung der auf das END-Kommando folgenden Anweisung.

Eine Variation des obigen Programmes kann man sinnvoll durch CONT erneut starten.

Beispiel:

```
10 PRINT "WOLLEN WIR BEGINNEN? JA/NEIN"
20 INPUT A$
30 IF A$="JA" THEN GOTO 50
40 END:PRINT "ANDERS UEBERLEGT?"
50 PRINT"ALSO LOS!"

RUN
WOLLEN WIR BEGINNEN? JA/NEIN
? NEIN

READY.
CONT
ANDERS UEBERLEGT?
ALSO LOS!

READY.
```

Anmerkung zur Kompatibilität: Bei manchen Computern (z. B. Schneider CPC 464/664) werden gespeicherte Tonfolgen auch nach Erreichen eines END-Kommandos im Programm weiter gespielt.

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  END
> COMMODORE VC20,64         :  END
> COMMODORE 16,116,PLUS4    :  END
> COMMODORE 128             :  END
> ENTERPRISE 64,128         :  END
    oder END Schleifentyp       Ende von Schleifen (IF,WHEN,...)
> MSX-STANDARD              :  END
> SCHNEIDER CPC464          :  END
> SCHNEIDER CPC664,CPC6128  :  END
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

ENT

Befehlsart: Anweisung

ENT **Folgenummer,Phasenparameter**

Beschreibung: Mit ENV (siehe dort!) läßt sich die Lautstärke eines Tones verändern. Mit ENT kann man die Tonhöhe verändern. Die Effekte sind dabei nicht ganz so deutlich wie bei der Änderung der Lautstärkenhüllkurve. Durch Variation der Tonhöhenhüllkurve erreichen wir zum Beispiel ein Vibrieren oder Hallen des Tons.

Die **Folgenummer** kann Werte zwischen 1 und 15 annehmen. Sie kennzeichnet die Tonveränderungsfolge. Ist sie negativ, wird der Formatteil **Phasenparameter** wiederholt. Er gibt die Veränderungsfolge an. Diese bestimmt, wie die Tonhöhe variiert wird.

Beispiel:

```
10 SOUND 1,478,50
20 SOUND 1,379,50
30 SOUND 1,319,50
40 ENT 1,3,4,5,1,2,3,1,-1,5
50 SOUND 1,478,50,,,1,0
60 SOUND 1,379,50,,,1,0
70 SOUND 1,319,50,,,1,0
80 GOTO 10
```

Anmerkung: Mit ENT und ENV lassen sich durch Versuchsfolgen Werte finden, mit denen der Computer bestimmte Musikinstrumente imitieren kann.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	ENT Folgenummer,Phasenparameter
› SCHNEIDER CPC664,CPC6128	:	ENT Folgenummer,Phasenparameter
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

ENTER

Befehlsart: Kommando

ENTER "**Programmname**"

Beschreibung: Das Programm mit dem **Programmnamen** wird von der Diskette oder von der Kassette geladen und im Rechnerspeicher abgelegt.
Der Befehl ENTER ähnelt also den Befehlen LOAD, CLOAD oder DLOAD (siehe jeweils dort), es gibt jedoch einige entscheidende Unterschiede:

- Falls sich bereits ein Programm im Speicher befindet, werden von dem mit ENTER geladenen Programm nur die Zeilen überschrieben, die die gleichen Zeilennummern aufweisen.
- Es lassen sich nur Programme laden, die mit dem LIST-Befehl (siehe dort) abgelegt wurden.

Mit dem **Programmnamen** wird durch C, D1 oder D2 angegeben, ob das Programm von der Kassette (**Programmname** = C:**Programmname**) oder von der Diskette (**Programmname** = D1:**Programmname**) geladen wird.

Beispiele:

ENTER "C:"

ENTER "D1:VERSUCH"

Anmerkung zur Kompatibilität: Das BASIC-Kommando ENTER ist nicht mit dem Drücken der ENTER-Taste beim Sinclair, bei den Schneider CPC und anderen Rechnern zu verwechseln.

Kompatibilitätsliste:

```
> ATARI 600,800,130XE        :  ENTER "Programmname"
  COMMODORE VC20,64          :
  COMMODORE 16,116,PLUS4     :
  COMMODORE 128              :
  ENTERPRISE 64,128          :  MERGE "Programmname"
  MSX-STANDARD               :  MERGE "CAS:Programmname"
  SCHNEIDER CPC464           :  MERGE "Programmname"
  SCHNEIDER CPC664,CPC6128   :  MERGE "Programmname"
  SINCLAIR ZX81              :
  SINCLAIR SPECTRUM          :  MERGE "Programmname"
                                MERGE*"M";Laufwerksnummer;"Programmname"
```

ENV

Befehlsart: Anweisung

ENV **Zahl 1,Zahl 2,Zahl 3,Zahl 4**

Beschreibung: Die Lautstärkenhüllkurve eines Tones wird verändert. Dabei bezeichnet **Zahl 1** die Nummer der definierten Hüllkurve, die Werte zwischen Ø und 15 annehmen kann. **Zahl 2** gibt an, in wie vielen Schritten die Lautstärke sich ändern soll. Hier lassen sich höchstens 127 Schritte eingeben. In **Zahl 3** legt man den Unterschied der einzelnen Lautstärken fest. Der Wert kann zwischen -128 und 127 liegen. **Zahl 4** definiert die Pausenzeit zwischen den in **Zahl 2** festgelegten Schritten. Die Grundzeit beträgt 1/1ØØ Sekunde und kann von Ø bis 255 dauern. Als Pausenzeit kann sie folglich bis zur nächsten Änderung also maximal 2,55 Sekunden umfassen. Der Wert ergibt sich aus 1/1ØØ*Zahl 4.

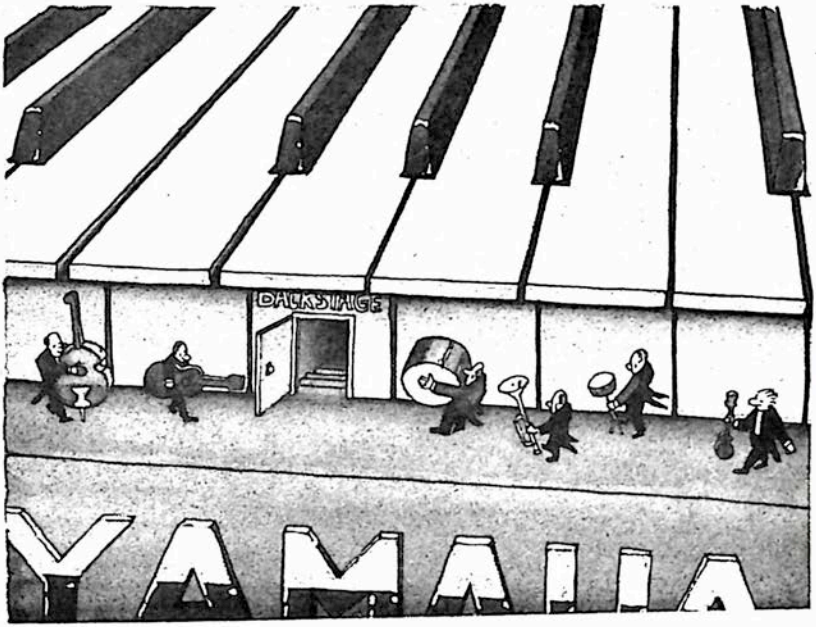
Beispiel:

```
10 ENV 1,5,2,200
20 SOUND 1,300,1000,5,1
```

Ein Ton (3ØØ) wird in diesem Beispiel mit der Hüllkurve Nummer 1 in 5 Schritten in seiner Lautstärke verändert, wobei die Lautstärke für 2Ø (2ØØ) die gleiche bleibt.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	ENVELOPE Zahlenliste
ENTERPRISE 64,128	:	ENVELOPE Zahlenliste
MSX-STANDARD	:	
> SCHNEIDER CPC464	:	ENV Zahlenliste
> SCHNEIDER CPC664,CPC6128	:	ENV Zahlenliste
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	



ENVELOPE

Befehlsart: Anweisung

ENVELOPE **Zahl 1,Zahl 2,Zahl 3,Zahl 4,Zahl 5,Zahl 6,Zahl 7**

Beschreibung: Mit ENVELOPE wird eine Hüllkurve für eine Tonfolge definiert. Der Befehl ähnelt der Anweisung ENV (siehe dort).

ENVELOPE

Es bedeutet

Zahl 1	Nummer der Hüllkurve (Ø bis 9)
Zahl 2	Anschlagzeit (Ø bis 15)
Zahl 3	Abschwellzeit (Ø bis 15)
Zahl 4	Haltezeit (Ø bis 15)
Zahl 5	Ausklingzeit (Ø bis 15)
Zahl 6	Wellenform: Ø Dreieck 1 Sägezahn 2 Rechteck 3 Rauschen 4 Ringmodulation
Zahl 7	Impulsbreite

Ist eine der Zahlen zwischen **Zahl 2** und **Zahl 7** gleich Null, kann sie weggelassen werden.

Beispiele:

```
ENVELOPE 1,3,4,5,6,2,2Ø48
ENVELOPE 1,2,,12,5,2,123
ENVELOPE 1
```

Anmerkung: Die Anwendung dieses Befehls setzt einige Programmiererfahrung im Sound-Bereich voraus.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
> COMMODORE 128	:	ENVELOPE Zahlenliste
> ENTERPRISE 64,128	:	ENVELOPE Zahlenliste
MSX-STANDARD	:	
SCHNEIDER CPC464	:	ENV Zahlenliste
SCHNEIDER CPC664,CPC6128	:	ENV Zahlenliste
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

EOF

Befehlsart: Funktion

EOF

Beschreibung: EOF bedeutet END OF FILE, also Ende einer Datei. Hat die Datei das Dateiende erreicht, erhält man bei der Abfrage dieser Funktion den Wert -1, ansonsten den Wert Ø.

Beispiel:

```
1Ø a= EOF
2Ø print a
```

Anmerkung zur Kompatibilität: Bei Rechnern, die nicht über diese Funktion verfügen, kann man beim Abspeichern einer Datei einen "unmöglichen" als letzten Wert abspeichern und dann beim Einlesen der Datei nach diesem Wert fragen - zum Beispiel über den IF-Befehl (siehe IF ...THEN). Sobald dieser Wert erreicht wurde, ist die Datei zu Ende.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
> MSX-STANDARD              : EOF(Dateinummer)
> SCHNEIDER CPC464          : EOF
> SCHNEIDER CPC664,CPC6128  : EOF
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

EPS

Befehlsart: Funktion

EPS(Zahl)

Beschreibung: Die Funktion gibt den kleinsten Wert zurück, der durch Addition oder Subtraktion eine Veränderung der **Zahl** im Speicher bewirkt.

Wenn die **Zahl** sehr groß ist, benötigt man einen großen Wert, um sie für den Rechner bemerkbar zu verändern. Ist die **Zahl** sehr klein, kann man schon mit kleinsten Werten eine Veränderung herbeiführen. Das liegt an der Rechengenauigkeit des Computers und an der Tatsache, daß der Rechner für Zahlen stets nur eine bestimmte Anzahl von Stellen vorsieht.

Beispiel:

```
PRINT EPS(1ØØØØØØØ)
 .ØØØ1
READY
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	EPS(Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

ER

Befehlsart: Variable

ER

Beschreibung: Die Variable enthält die Nummer des zuletzt vom BASIC-Interpreter entdeckten Fehlers.
Die Fehlermeldungen sind nach dem Handbuch zu entschlüsseln.

Beispiele:

```
I=ER

PRINT ER
```

Anmerkung: Die Wirkung von ER ist ähnlich der von ERR (siehe dort).

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  ER
> COMMODORE 128             :  ER
  ENTERPRISE 64,128         :  EXTYPE
  MSX-STANDARD              :  ERR
  SCHNEIDER CPC464          :  ERR
  SCHNEIDER CPC664,CPC6128  :  ERR
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

ERASE

Befehlsart: Kommando

ERASE **Variablenname**

Beschreibung: Ein durch den DIM-Befehl (siehe DIM) definiertes Feld wird gelöscht. Dadurch schafft man freien Speicherplatz.

Beispiel:

```
ERASE a$,b
```

Anmerkungen zur Kompatibilität: Das ERASE-Kommando beim Sinclair Spectrum hat ein anderes Format: ERASE **Device-Bezeichnung, Zahl, Dateiname**

Die Device-Bezeichnung kann nur "M" oder "m" sein. Die **Zahl** bezeichnet die Nummer des Microdrive und kann einen Wert zwischen 1 und 8 annehmen.

Die Datei mit dem **Dateinamen** wird mit diesem Befehl gelöscht.

Bei Rechnern, die den ERASE-Befehl nicht besitzen, gibt es in BASIC keine Möglichkeit zur Umschreibung.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	ERASE Variablenname
› SCHNEIDER CPC464	:	ERASE Variablenname
› SCHNEIDER CPC664,CPC6128	:	ERASE Variablenname
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

ERL

Befehlsart: Variable

ERL

Beschreibung: Die Variable ERL enthält die Zeilennummer des Programmes, in welcher der Computer einen Programmierfehler gefunden hat.

Wurde - wie im untenstehenden Beispiel - das NEXT einer FOR...TO-NEXT-Schleife vergessen, so gibt die Variable ERL die Zeilennummer wieder, in der die Schleife beginnt.

Die Variable ist in Verbindung mit ON ERROR GOTO (siehe dort!) und ERR (siehe dort!) geeignet, eigene Fehlermeldungen mit Angabe der Fehlerzeile zu schreiben.

Beispiel:

```
10 ON ERROR GOTO 100
20 FOR i=1 TO 26
30 PRINT CHR$(64+i);
50 END
100 PRINT "fehler in zeile" ERL
110 LIST
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :  EL
  COMMODORE 128             :  EL
  ENTERPRISE 64,128         :  EXLINE
> MSX-STANDARD              :  ERL
> SCHNEIDER CPC464          :  ERL
> SCHNEIDER CPC664,CPC6128  :  ERL
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

ERR

Befehlsart: Variable

ERR

Beschreibung: Die Variable ERR enthält beim Auftreten eines Programmfehlers, wie zum Beispiel SYNTAX ERROR, NEXT MISSING usw., die Fehlernummer. Mit Hilfe der Fehlermeldungsliste im Handbuch kann man die Art des Fehlers ermitteln.

In Verbindung mit ON ERROR GOTO (siehe dort!) und ERL (siehe dort!) lassen sich damit eigene Fehlermeldungen schreiben. Unter anderem sind auch deutsche Fehlermeldungstexte möglich.

Beispiel:

```
10 ON ERROR GOTO 100
20 FOR i=1 TO 26
30 PRINT CHR$(i+64);
40 :
50 END
100 PRINT"fehler"ERR"in zeile"ERL
110 LIST
```

Das Beispiel gibt als Ergebnis Fehlermeldung 26 zurück. Die Fehlermeldung lautet NEXT MISSING. Außerdem erscheint die Zeile 2Ø; in dieser Zeile beginnt die Schleife, zu der das NEXT fehlt.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	ER
COMMODORE 128	:	ER
ENTERPRISE 64,128	:	EXTYPE
› MSX-STANDARD	:	ERR
› SCHNEIDER CPC464	:	ERR
› SCHNEIDER CPC664,CPC6128	:	ERR
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

ERR$

Befehlsart: Funktion

ERR$(**Zahl**)

Beschreibung: Diese Funktion übergibt einer Stringvariablen den zur **Zahl** gehörigen Fehlertext. Die **Zahl** darf dabei zwischen 1 und 127 liegen und soll ganzzahlig sein.

Mit Hilfe der ER-Variablen (siehe dort) kann man ein Unterprogramm zur Fehlerbehandlung schreiben. Der Text der Fehlermeldung erscheint in englischer Sprache.

Diese Funktion kann nur in Verbindung mit dem TRAP-Befehl (siehe dort) verwendet werden.

Beispiel:

```
0 TRAP 60000
.
. gebe hier bitte irgendein kleines fehlerhaftes Programm ein
.
60000 REM FEHLERBEHANDLUNG
60010 ZE=EL: REM FEHLERZEILE FESTHALTEN
60020 FE=ER: REM FEHLERART FESTHALTEN
60030 FE$=ERR$(FE): REM FEHLERTEXT ERMITTELN
60040 REM FEHLERAUSGABE
60050 PRINT"DU HAST LEIDER EINEN FEHLER GEMACHT!"
60060 PRINT"ICH HABE EINEN FEHLER IN DER ZEILE"ZE
60070 PRINT"FESTGESTELLT. ER LAUTET:"
60080 PRINTFE$
60090 PRINT"VERBESSERE IHN DOCH GLEICH!"
60100 END
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	ERR$(Zahl)
› COMMODORE 128	:	ERR$(Zahl)
ENTERPRISE 64,128	:	EXSTRING$(Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

ERROR

Befehlsart: Kommando

ERROR **Zahl**

Beschreibung: Anhand dieses Befehls kann man einen Programmabbruch mit Fehlermeldung bewirken, obwohl kein Programmierfehler vorliegt. Welchen Fehler der Computer meldet, hängt von der **Zahl** ab. Fehlermeldungen sind möglich als ganzzahlige Werte zwischen 1 und 3Ø. Um welche Fehlermeldungen es sich jeweils handelt, kann der Fehlermeldungsliste des Handbuches entnommen werden.

Bei einer **Zahl** zwischen 31 und 255 meldet der Rechner "UNKNOWN ERROR in Zeile...". Wird eine Zahl größer als 255 eingegeben, erfolgt als Fehlermeldung "IMPROPER ARGUMENT".

Beispiel:

```
1Ø INPUT A
2Ø IF A=2 THEN ERROR 1 ELSE PRINT "aaaa"
3Ø GOTO 1Ø
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64;128	:	CAUSE EXCEPTION Zahl
› MSX-STANDARD	:	ERROR Zahl
› SCHNEIDER CPC464	:	ERROR Zahl
› SCHNEIDER CPC664,CPC6128	:	ERROR Zahl
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

EVERY GOSUB

Befehlsart: Anweisung

EVERY **Zahl 1,Zahl 2** GOSUB **Zeilennummer**

Beschreibung: Alle **Zahl1*Ø.Ø2** Sekunden wird in ein Unterprogramm gesprungen, das in Zeile **Zeilennummer** beginnt.

Das Programm wird wärenddessen normal abgearbeitet, bis die Zeit (**Zahl1*Ø.Ø2**) abgelaufen ist. Darauf wird sofort das Unterprogramm ausgeführt, danach das Hauptprogramm weiter abgearbeitet.

Die **Zahl 2** spricht eine der vier Uhren im Schneider-Computer an. Sie kann deshalb nur die Werte Ø,1,2 oder 3 annehmen. Läßt man das Komma und die **Zahl 2** im Format des Befehls weg, so wird automatisch die Uhr Ø angewählt.

Die **Zeilennummer** gibt die Startzeile (auch Anfangsadresse genannt) des Unterprogramms an. Das Unterprogramm ist mit dem RETURN-Befehl zu schließen (siehe GOSUB).

Der EVERY...GOSUB-Befehl ist dem Befehl AFTER...GOSUB (siehe dort!) verwandt; hier wird allerdings immer wieder in das Unterprogramm verzweigt, während dies bei AFTER... GOSUB nur einmal geschieht.

Bei MSX-Computern heißt dieser Befehl ON INTERVALL GOSUB (siehe dort!).

Beispiel:

```
10 EVERY 100 GOSUB 40
20 PRINT "BUECHER LESEN MACHT SPASS!"
30 GOTO 20
40 PRINT:PRINT "SCHNEIDER-";:PRINT :PRINT
50 RETURN
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	ON INTERVAL-Zahl1 GOSUB Zeilennummer
› SCHNEIDER CPC464	:	EVERY Zahl1,Zahl2 GOSUB Zeilennummer
› SCHNEIDER CPC664,CPC6128	:	EVERY Zahl1,Zahl2 GOSUB Zeilennummer
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

EXIT

Befehlsart: Anweisung

EXIT

Beschreibung: Mit diesem Befehl kann man die DO-Schleife (siehe DO) verlassen. Sinnvolle Anwendungen des Befehls ergeben sich zum Beispiel in Verbindung mit einer IF-Bedingung (siehe IF).

Beispiel:

```
10 DO
20 I=I+1
30 PRINT I
40 IF I=40 THEN EXIT
50 LOOP
60 END
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  EXIT
> COMMODORE 128             :  EXIT
> ENTERPRISE 64,128         :  EXIT DO
                               EXIT DEF
                               EXIT FOR
                               EXIT HANDLER
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

EXLINE

Befehlsart: Funktion

EXLINE

Beschreibung: Die Funktion gibt die Zeilennummer wieder, in der zuletzt ein Fehler aufgetreten ist (siehe dazu auch EL und ERL).

Ist kein Fehler aufgetreten, wird der Wert Ø zurückgegeben.

Beispiel:

```
100 PRINT "WOLLEN WIR EIN SPIEL MACHEN?"
110 PRINT "GIB BITTE >JA< ODER >NEIN< EIN."
120 INPUT AN$
130 IF AN$="JA" THEN LOAD "GRAND PRIX",8
140 IF AN$=NEIN THEN PRINT "SCHADE!":CLS:END
1000 PRINT EXLINE
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	EL
COMMODORE 128	:	EL
› ENTERPRISE 64,128	:	EXLINE
MSX-STANDARD	:	ERL
SCHNEIDER CPC464	:	ERL
SCHNEIDER CPC664,CPC6128	:	ERL
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

EXP

Befehlsart: Funktion

EXP(**Zahl**)

Beschreibung: Der Befehl gibt den Wert von e potenziert mit der **Zahl** wieder. e ist dabei der natürliche Logarithmus von 1, das heißt angenähert 2.7182818.

Beispiel:

```
PRINT EXP (1)
 2.7182818
```

Anmerkung: Die Funktion ist nur sinnvoll anzuwenden, wenn man mit der Exponentialrechnung vertraut ist.

Kompatibilitätsliste:

› ATARI 600,800,130XE	: EXP(Zahl)
› COMMODORE VC20,64	: EXP(Zahl)
› COMMODORE 16,116,PLUS4	: EXP(Zahl)
› COMMODORE 128	: EXP(Zahl)
› ENTERPRISE 64,128	: EXP(Zahl)
› MSX-STANDARD	: EXP(Zahl)
› SCHNEIDER CPC464	: EXP(Zahl)
› SCHNEIDER CPC664,CPC6128	: EXP(Zahl)
› SINCLAIR ZX81	: EXP Zahl
› SINCLAIR SPECTRUM	: EXP Zahl

EXSTRING$

Befehlsart: Funktion

EXSTRING$(**Zahl**)

Beschreibung: Die Funktion gibt den Fehlermeldungstext zum Fehler wieder, der die **Zahl** als Nummer hat.

Beispiel:

```
PRINT EXSTRING$(1ØØØ)
 UNERWARTETER WERT EINGEGEBEN
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :  ERR$(Zahl)
  COMMODORE 128             :  ERR$(Zahl)
> ENTERPRISE 64,128         :  EXSTRING$(Zahl)
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

EXT

Befehlsart: Kommando

EXT **"String"**
:String

Beschreibung: Mit diesem Kommando lassen sich Programme aufrufen, die sich innerhalb des Rechnerbetriebssystems befinden. (Das Betriebssystem ist jener Bestandteil des Rechners, der für Ein- und Ausgabe zuständig ist.)

Der String gibt den Namen der Routine an, die aufgerufen werden soll.
Mit String = help erhält der Benutzer die Namen aller Funktionen, die ihm beim EXT-Befehl zur Verfügung stehen.

Beispiel:

```
EXT "BRD"
```

Dieser Befehl schaltet den Rechner auf die Verwendung des deutschen Zeichensatzes um.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	EXT "String" :String
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

EXTYPE

Befehlsart: Funktion

EXTYPE

Beschreibung: Diese Funktion gibt die Nummer des zuletzt entdeckten Fehlers an. Ist kein Fehler aufgetreten, wird 2ØØ1 zurückgegeben (siehe auch ERR und ER).

Beispiele:

```
PRINT EXTYPE
A=EXTYPE
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	ER
COMMODORE 128	:	ER
› ENTERPRISE 64,128	:	EXTYPE
MSX-STANDARD	:	ERR
SCHNEIDER CPC464	:	ERR
SCHNEIDER CPC664,CPC6128	:	ERR
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

FAST

Befehlsart: Kommando

FAST

Beschreibung: Dieser Befehl schaltet den Rechner vom langsamen Verarbeitungsmodus (1 MHz) auf den schnellen (2 MHz). Das Herzstück des Rechners arbeitet somit doppelt so schnell wie im Normalfall nach dem Einschalten des Computers.

In diesem Modus kann man nur noch mit einem 8Ø-Zeichen-Monitor arbeiten. Verwendet man einen 4Ø-Zeichen-Monitor oder einen Fernsehapparat, läßt sich in diesem Modus nicht mehr arbeiten.

Der Befehl mit der entgegengesetzten Wirkung, der den Rechner vom schnellen auf den langsamen Verarbeitungsmodus umschaltet, lautet SLOW (siehe dort).

Beispiel: FAST

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
› COMMODORE 128	:	FAST
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

FETCH

Befehlsart: Kommando

FETCH **Zahl 1**,**Adresse 1**,**Zahl 2**,**Adresse 2**

Beschreibung: Eine bestimmte Anzahl von Bytes wird aus einer bestimmten Speicherbank an einer bestimmten Adresse im BASIC-Speicher "abgeholt" (engl. "to fetch" = holen).

Zahl 1 gibt die Anzahl der Bytes an, die in den BASIC-Speicher übertragen werden sollen. Der Wert liegt zwischen Ø und 65535.

Die **Adresse 1** bezeichnet die Speicheradresse, ab der die übertragenen Bytes im BASIC-Speicher abgelegt werden sollen.

Zahl 2 gibt die Speicherbank an, von der die Bytes geholt werden sollen (siehe dazu auch BANK). Der Wert liegt zwischen Ø und 7.

Die **Adresse 2** kennzeichnet jene Speicheradresse der durch **Zahl 2** bestimmten Bank, ab der die Bytes übertragen werden sollen. Dieser Wert liegt zwischen Ø und 65535.

Beispiel:

FETCH 1Ø23,4Ø96,7,2Ø48

Mit diesem Format des Befehls werden ab der Speicherzelle 2Ø48 1Ø24 Bytes aus der Speicherbank 7 abgeholt und in den BASIC-Speicher ab der Speicherzelle 4Ø96 übertragen.

Anmerkung: Dieser Befehl benötigt eine gewisse Erfahrung beim Programmieren, da sich sehr leicht Fehler einschleichen.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  FETCH Zahl1,Adresse1,Zahl2,Adresse2
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

FILTER

Befehlsart: Anweisung

FILTER **Zahl 1,Zahl 2,Zahl 3,Zahl 4,Zahl 5**

Beschreibung: Mit diesem Befehl kann man die Klänge, die ein Computer erzeugt, beeinflussen. Er erfordert allerdings einige Erfahrung im Programmieren von Musik-Elementen.

Mit der **Zahl 1** wird die Filterfrequenz festgelegt. Der entsprechende Wert ist ganzzahlig und liegt zwischen Ø und 2Ø47.

Die reale Frequenz erhält man durch die Formel

Frequenz=(**Zahl** 1-3Ø)/5.8

Diese Formel unterliegt von Rechnertyp zu Rechnertyp leichten Abweichungen.

Zahl 2 legt fest, ob man den Tiefpaßfilter verwendet (1) oder nicht (Ø).

Zahl 3 legt fest, ob man den Bandpaßfilter verwendet (1) oder nicht (Ø).

Zahl 4 legt fest, ob man den Hochpaßfilter verwendet (1) oder nicht (Ø).

Verwendet man mehrere der Filter, so überlagern sich diese. Falls man eine der **Zahl**en nicht verwendet, wird der jeweils vorher gültige Wert angenommen.

Mit der **Zahl 5** läßt sich die Resonanz festlegen. Auch dieser Wert ist ganzzahlig und liegt zwischen Ø und 15.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  FILTER Zahl1,Zahl2,Zahl3,Zahl4,Zahl5
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

FIX

Befehlsart: Funktion

FIX(**Zahl**)

Beschreibung: Durch den Befehl FIX werden die Nachkommastellen der **Zahl** abgeschnitten. Man erhält also eine ganze Zahl.

Beispiel:

```
10 FOR I =-3 TO 3 STEP 0.7777
20 PRINT I;" WIRD ZU ";FIX(I)
30 NEXT I
RUN

-3  WIRD ZU -3
-2.2223  WIRD ZU -2
-1.4446  WIRD ZU -1
-0.666900001  WIRD ZU  0
 0.110799999  WIRD ZU  0
 0.8885  WIRD ZU  0
 1.6662  WIRD ZU  1
 2.4439  WIRD ZU  2
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :  IP(Zahl)
› MSX-STANDARD              :  FIX(Zahl)
› SCHNEIDER CPC464          :  FIX(Zahl)
› SCHNEIDER CPC664,CPC6128  :  FIX(Zahl)
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

FKEY

Befehlsart: Kommando

SET #**Kanalnummer**:FKEY **Zahl String**

Beschreibung: Mit diesem Befehl kann der Benutzer die Funktionstasten neu belegen (siehe auch unter KEY).
Im Format des Befehls gibt die **Zahl** die Nummer der Funktionstaste an, die neu belegt werden soll. Mögliche Werte sind also 1 bis 16. Der **String** gibt den gewünschten Funktionstasteninhalt an. Mit der **Kanalnummer** legt man fest, über welchen Kanal der Befehl ausgeführt wird. Normalerweise ist dies Nummer 1Ø5.

Beispiel:

```
10 FOR I=1 TO 16
20 SET#105:FKEY I "HEUTE NICHT"
30 NEXT
```

Anmerkung: Mit FKEY CLEAR kann man diese Neubelegung der Funktionstasten wieder löschen (siehe auch unter CLEAR).

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :  KEY Zahl,String
  COMMODORE 128             :  KEY Zahl,String
> ENTERPRISE 64,128         :  SET #Kanal:FKEY Zahl String
  MSX-STANDARD              :  KEY Zahl,String
  SCHNEIDER CPC464          :  KEY Zahl,String
  SCHNEIDER CPC664,CPC6128  :  KEY Zahl,String
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

FLASH

Befehlsart: Anweisung

FLASH **Zahl**

Beschreibung: Zeichen werden auf dem Bildschirm zum Blinken gebracht.

Ist die **Zahl** gleich 1, so wird der Blinkmodus eingeschaltet, ist sie Ø, wird er abgeschaltet. Die Eingabe einer anderen Zahl ist nicht möglich. Die jeweilige Veränderung wird so lange beibehalten, bis ein neuer FLASH-Befehl im Programm erreicht wird.

Beispiel:

```
10 PRINT "*";
20 FLASH 1
30 PRINT "*";
40 FLASH 0
50 FOR T=1 TO 300:NEXT
60 GOTO 10

READY.
```

Anmerkung zur Kompatibilität: Derselbe Effekt kann bei anderen Rechnern durch anderslautende produkteigene Befehle erzielt werden - bei den Schneider CPC-Computern anhand des INK-Befehls (siehe dort).

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
› SINCLAIR SPECTRUM         :  FLASH Zahl
```

FLUSH

Befehlsart: Anweisung

FLUSH #**Kanalnummer**

Beschreibung: Die Daten, die in dem durch die **Kanalnummer** festgelegten Kanal vorhanden sind, werden ausgesendet. Der Kanal wird hierbei nicht geschlossen.

Beispiel:

FLUSH #1Ø9

Anmerkung: Dieser Befehl ist in erster Linie für die Verwendung von **Computer-Netzwerken** interessant.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
› ENTERPRISE 64,128         :  FLUSH #Zahl
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

FN

Befehlsart: Funktion

FNFunktionsname(Zahl)

Beschreibung: Der Befehl FN ruft eine mit DEF FN (siehe dort!) definierte Funktion auf.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE      :
> COMMODORE VC20,64        :  FNFunktionsname(Zahl)
> COMMODORE 16,116,PLUS4   :  FNFunktionsname(Zahl)
> COMMODORE 128            :  FNFunktionsname(Zahl)
  ENTERPRISE 64,128        :  CALL Funktionsname(Zahl)
> MSX-STANDARD             :  FNFunktionsname(Zahl)
> SCHNEIDER CPC464         :  FNFunktionsname(Zahl)
> SCHNEIDER CPC664,CPC6128 :  FNFunktionsname(Zahl)
  SINCLAIR ZX81            :
> SINCLAIR SPECTRUM        :  FN Funktionsname(Zahl)
```

FOR...TO...STEP

Befehlsart: Anweisung

FOR **Variablenname**=**Zahl 1**TO**Zahl 2** STEP **Zahl 3**

Beschreibung: In einer gezählten Schleife wird eine Anweisung oder werden mehrere Anweisungen mehrere Male ausgeführt. Die Differenz zwischen **Zahl 2** und **Zahl 1** bestimmt dabei, wie oft diese Anweisung ausgeführt wird, bzw. wie oft diese Anweisungen ausgeführt werden. Jedes Durchlaufen der Schleife wird mit NEXT (siehe NEXT) abgeschlossen.

Der **Variablenname** gibt die sogenannte Laufvariable an. Sie wird mit jedem Durchlaufen der Schleife gezählt, in der ein vorgegebener Programmteil abgearbeitet wird, und zwar vom Startwert **Zahl 1** bis zum Endwert **Zahl 2** in der

Schrittweite **Zahl 3**. Ist keine Schrittweite angegeben, wird automatisch der Wert 1 angenommen.

Die gezählte Schleife gehört zu den wichtigsten BASIC-Anweisungen und läßt sich auf sehr verschiedene Arten anwenden. Fehlt zum Beispiel in der Schleife die Anweisung oder der vorgegebene Programmteil, so dient die Schleife lediglich zur Verzögerung, etwa vor dem Löschen des Bildschirms.

Beispiel:

```
10 PRINT "␣GOUTEN TAG!"
20 FOR T=1 TO 2000:NEXT
30 PRINT "␣"

READY.
```

Das Programm im folgenden Beispiel zählt alle Zahlen von 1 bis 20 und bringt sie auf den Bildschirm.

```
10 FOR A=1TO20
20 PRINT A;
30 NEXT A

RUN
 1  2  3  4  5  6  7  8  9  10  11  12
13  14  15  16  17  18  19  20
READY.
```

Im folgenden Beispiel geschieht dasselbe, jedoch in Zweierschritten:

```
10 FOR A=1TO20 STEP2
20 PRINT A;
30 NEXT A

RUN
 1  3  5  7  9  11  13  15  17  19
READY.
```

Die Schrittweite kann man auch als negative Zahl angeben, sofern der Startwert (**Zahl 1**) größer ist als der Endwert (**Zahl 2**).

```
10 FOR A=20TO1 STEP-2
20 PRINT A;
30 NEXT A

RUN
 20  18  16  14  12  10  8  6  4  2
READY.
```

Anmerkung: Innerhalb einer Schleife dürfen wiederum Schleifen verwendet werden. Dabei sind folgende Regeln zu beachten:

- Für jede FOR-Anweisung muß eine NEXT-Anweisung vorhanden sein und umgekehrt. Andernfalls sind Fehlermeldungen wie

  ?NEXT WITHOUT FOR ERROR

oder

  UNEXPECTED NEXT IN 12Ø

die Folge.

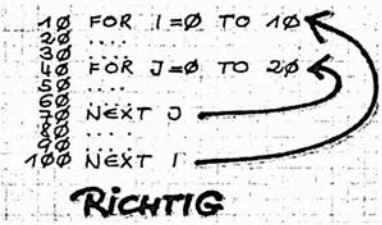
- Die gezählte Variable in jeder Schleife muß einen unterschiedlichen Variablennamen erhalten.
- Jede in einer Schleife eingefügte Schleife muß innerhalb dieser abgeschlossen sein.

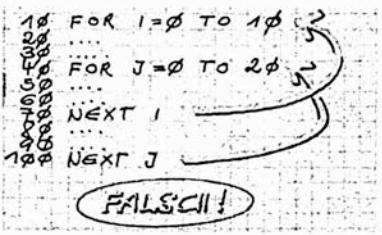
Theoretisch ist die Zahl der möglichen verschachtelten Schleifen unbegrenzt. Die Grenzen nach oben sind programmtechnischer Art oder werden durch die Speicherkapazität des Computers vorgegeben.
- Es gibt eine Reihe von Kommandos und Anweisungen, die innerhalb einer FOR...TO...STEP-NEXT-Schleife nicht verwendet werden können (zum Beispiel CLR).

Beispiel:

```
110 FOR I=1 TO 20
120 R=8
130 FOR S=1 TO R-4
140 READ X$
150 PRINT X$,
160 NEXT S
170 PRINT:PRINT I:PRINT
180 RESTORE: NEXT I
190 DATA ICH,BIN,EINE,"SCHLEIFE:"
```

Statt eines numerischen Wertes kann man für **Zahl 1**, **Zahl 2** oder **Zahl 3** auch eine Variable oder einen Rechenausdruck einsetzen.





Beispiel:

```
10 FOR I=1 TO 23
20 READ A$
30 PRINT" ";A$;
40 FOR K=1 TO 800
50 NEXT K
60 NEXT I
70 DATAICH,HOLE,MIR,DIE,INFORMATIONEN,AUS
80 DATA DEM,"DATA-",ZEILEN.,IN,DER,MITTE,MEINER,SCHLEIFE,HABE,ICH,NOCH,EINE
90 DATA "SCHLEIFE,",DIE,ZUR,VERZOEGERUNG,DIENT.
```

Anmerkung zur Kompatibilität: Mit dem Befehl EXIT FOR kann man bei den Enterprise-Rechnern die FOR-Schleife vorzeitig verlassen (siehe EXIT).

Kompatibilitätsliste:

```
› ATARI 600,800,130XE        :  FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› COMMODORE VC20,64          :  FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› COMMODORE 16,116,PLUS4     :  FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› COMMODORE 128              :  FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› ENTERPRISE 64,128          :  FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› MSX-STANDARD               :  FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› SCHNEIDER CPC464           :  FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› SCHNEIDER CPC664,CPC6128   :  FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› SINCLAIR ZX81              :  FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› SINCLAIR SPECTRUM          :  FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
```

FORMAT

Befehlsart: Kommando

FORMAT **Option**

Beschreibung: Mit diesem Befehl werden die Schnittstellen des SINCLAIR SPECTRUM initialisiert. Mit Hilfe der **Option** kann man eine Diskette formatieren sowie auf die serielle Schnittstelle einstellen.

FORMAT "M";**Zahl**;**Name**

formatiert die Diskette in dem Laufwerk, das mit der **Zahl** gekennzeichnet ist und weist dieser den **Name**n zu (siehe auch HEADER).

FORMAT "N";**Zahl**

weist dem Rechner bei Netzwerkbetrieb die durch die **Zahl** bezeichnete Nummer (1 bis 64) zu. (Im Netzwerkbetrieb sind mehrere Rechner zu einem Netz zusammengeschlossen und können miteinander kommunizieren.)

Beispiel:

FORMAT "T",3ØØ

Dieser Befehl bewirkt eine Initialisierung der seriellen Schnittstelle für Textübertragung mit 3ØØ Baud.

Anmerkung: Alle Möglichkeiten dieses Befehls aufzuführen, würde den Rahmen dieses Referenz-Lexikons sprengen.

Anmerkung zur Kompatibilität: Der FORMAT-Befehl des SINCLAIR SPECTRUM ist dem Befehl SET des ENTERPRISE sehr ähnlich.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :  SET OPTION Parameter
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
> SINCLAIR SPECTRUM         :  FORMAT OPTION Parameter
```

FP

Befehlsart: Funktion

FP(**Zahl**)

Beschreibung: Die Funktion gibt die Nachkommastellen der **Zahl** zurück.

Beispiele:

```
PRINT FP(1.23)
 .23
READY

PRINT FP(-3.45)
 -.45
READY
```

Anmerkung zur Kompatibilität: Die Funktion kann bei Rechnern, die nicht über diesen BASIC-Befehl verfügen, simuliert werden:

FP(**Zahl**)

entspricht dann

SGN(Zahl)*(Zahl/-INT(ABS(Zahl)))

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  FP(Zahl)
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

FRE

Befehlsart: Funktion

FRE(**Zahl**)

Beschreibung: Durch diese Funktion wird die Größe des gesamten von BASIC nicht benutzten Speicherplatzes zurückgegeben. Die **Zahl** hinter der Funktion steht immer in Klammern. Sie hat auf die Ausgabe keinen Einfluß, darf aber nicht fehlen.
Man kann auch Buchstaben verwenden.

Im Direktmodus gibt die Funktion den freien Speicherplatz in Bytes an.

Beispiel:

```
PRINT FRE(0)
 29276
READY.
PRINT FRE(89)
 29276
READY.

PRINT FRE (SPEICHER)
 29276
READY.
```

Innerhalb eines Programmes läßt sich die FRE-Funktion auch zur Steuerung verwenden.

Beispiel:

```
4590 IF FRE(0)<4000 THEN GOSUB 6000
:
:
:
:
6000 REM *UNTERPROG MINIMAL SPEICHER*
```

Anmerkung zur Kompatibilität: Bei manchen BASIC-Dialekten (z. B. Schneider CPC 464/664) gibt es die Sonderform

```
FRE("")
```

Durch sie kann man auch noch kleinste freie Speicherplätze vor der Ausgabe sammeln.

Mit

```
FRE (X$)
```

wird der freie Speicherplatz für Zeichenketten angegeben. Bei manchen Computern (z. B. IBM P.C.) bewirkt dieses Format vor der Ausgabe eine Neuorganisation der Daten.

Kompatibilitätsliste:

›	ATARI 600,800,130XE	: FRE(Zahl)
›	COMMODORE VC20,64	: FRE(Zahl)
›	COMMODORE 16,116,PLUS4	: FRE(Zahl)
›	COMMODORE 128	: FRE(Zahl)
	ENTERPRISE 64,128	: FREE
›	MSX-STANDARD	: FRE(Zahl)
›	SCHNEIDER CPC464	: FRE(Zahl)
›	SCHNEIDER CPC664,CPC6128	: FRE(Zahl)
	SINCLAIR ZX81	:
	SINCLAIR SPECTRUM	:

FREE

Befehlsart: Funktion

FREE

Beschreibung: Die Funktion gibt die Größe des freien Speicherplatzes wieder zurück, der noch für das augenblicklich im Rechnerspeicher geladene Programm zur Verfügung steht. Der wiedergegebene Wert muß nicht immer derselbe sein wie der, den man durch den Befehl INFO (siehe dort) erhält.

Beispiele:

```
PRINT FREE
 44221
READY

a=FREE:PRINT a
```

Anmerkung: Man kann die Funktion FREE im Gegensatz zu INFO auch innerhalb eines BASIC-Programmes verwenden.

Anmerkung zur Kompatibilität: Die FREE-Funktion des ENTERPRISE entspricht dem FRE-Befehl (siehe dort) bei anderen Rechnern.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	FRE(Zahl)
COMMODORE VC20,64	:	FRE(Zahl)
COMMODORE 16,116,PLUS4	:	FRE(Zahl)
COMMODORE 128	:	FRE(Zahl)
ENTERPRISE 64,128	:	FREE
MSX-STANDARD	:	FRE(Zahl)
SCHNEIDER CPC464	:	FRE(Zahl)
SCHNEIDER CPC664,CPC6128	:	FRE(Zahl)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

GET

Befehlsart: Anweisung

GET **Stringvariable**
GET **Zahlenvariable**

Beschreibung: Zur Eingabe von Daten gibt es in den meisten BASIC-Dialekten außer dem INPUT-Befehl (siehe dort) auch diesen Befehl. Mit GET oder INKEY$ (siehe dort) kann man im Gegensatz zu INPUT jede beliebige Taste bzw. die dazugehörigen Zeichen eingeben, also auch Leerzeichen, Komma, Doppelpunkt usw.

Bei der GET-Anweisung holt sich der Computer die Informationen Stück für Stück in den Speicher. Eine Betätigung von RETURN, ENTER usw. ist nicht nötig, und auf dem Bildschirm erscheint kein Fragezeichen. Das Programm wird nicht unterbrochen, wie sich an dem folgenden kurzen Beispielprogramm zeigen läßt:

Beispiel:

```
10 GET A
20 PRINT A;:FOR T=1 TO 100:NEXT
30 GOTO 10

READY.
```

Das Programm verlangt in einem durch die Zeitschleife in Zeile 2Ø vorgegebenen Rhythmus die Eingabe von Zahlen. Wird keine Taste betätigt, schreibt das Programm die Ø. Will man erreichen, daß auch mit dem GET-Befehl auf eine Eingabe gewartet wird, muß die Zeile 1Ø folgendermaßen verändert werden:

```
10 GET A:IF A=0 THEN 10
20 PRINT A;
30 GOTO 10

READY.
```

GET

Ähnlich wie bei der Verwendung von numerischen Variablen kann man mit alphanumerischen Variablen arbeiten.

Beispiel:

```
10 GET A$
20 PRINT A$;" ";
30 GOTO 10

READY.
```

Auch in diesem Fall kann mit einer Bedingung das Fortlaufen des Programmes verhindert werden, bis eine Eingabe gemacht wurde. Eine Programmzeile
1Ø GET A$:IF A$:''''THEN 1Ø
bedeutet, daß das Programm so lange auf Zeile 1Ø zurückkehrt, bis eine Eingabe erfolgt ist. Auch die Leertaste wird in diesem Fall als Eingabe akzeptiert.

Beispiel:

```
10 GET A$:IF A$="" THEN 10
20 PRINT A$;:FOR T=1 TO 100:NEXT
30 GOTO 10

READY.
```

Das folgende kleine Programm verwandelt Computer und Drukker in eine "Schreibmaschine":

```
5 OPEN 1,5
10 GET A$
20 PRINT#1, A$;
30 GOTO 10

READY.
```

Beispiel für C 64 und EPSON FX

Anmerkung: Der Programmierer sollte bei der Anwendung dieses Befehles immer eine Stringvariable verwenden, da bei

Verwendung der Zahlenvariable ein Fehler auftritt, sobald man z. B. eine Buchstabentaste drückt. Wenn er mit Zahlen arbeitet, kann er den Befehl GET A wie folgt umschreiben:

```
1ØØ GET A$ : IF A$  "Ø" OR A$  "9" THEN 1ØØ
1ØØ A = VAL(A$)
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
› COMMODORE VC20,64         :  GET Stringvariable
                               GET Zahlenvariable
› COMMODORE 16,116,PLUS4    :  GET Stringvariable
                               GET Zahlenvariable
› COMMODORE 128             :  GET Stringvariable
                               GET Zahlenvariable
› ENTERPRISE 64,128         :  GET Stringvariable
  MSX-STANDARD              :  Stringvariable=INKEY$
  SCHNEIDER CPC464          :  Stringvariable=INKEY$
  SCHNEIDER CPC664,CPC6128  :  Stringvariable=INKEY$
  SINCLAIR ZX81             :  LET Stringvariable=INKEY$
  SINCLAIR SPECTRUM         :  LET Stringvariable=INKEY$
```

GET#

Befehlsart: Anweisung

GET# **Zahl 1**

Beschreibung: Der Befehl hat in den verschiedenen BASIC-Dialekten unterschiedliche Bedeutungen und unterschiedliche Formate.

Benutzt wird der Befehl in Verbindung mit einer vorher durch OPEN (siehe dort) eröffneten Datei mit der logischen File-Nummer **Zahl 1**. Aus dieser Datei werden Zahlen oder Zeichen einzeln in den Rechnerspeicher geholt.

Beispiel:

```
10 OPEN 1,5
20 GET# 1,A$

READY.
```

Anmerkung zur Kompatibilität: Die Werte, die **Zahl 1** annehmen kann, sind von Computertyp zu Computertyp sehr verschieden.

Für die Commodore-Computer hat der Befehl das Format

GET#Zahl,Variable

Beispiel:

GET#1,A$

Kompatibilitätsliste:

› ATARI 600,800,130XE	: GET #Zahl,Variable
› COMMODORE VC20,64	: GET #Zahl,Variable
› COMMODORE 16,116,PLUS4	: GET #Zahl,Variable
› COMMODORE 128	: GET #Zahl,Variable
› ENTERPRISE 64,128	: GET #Zahl,Stringvariable
MSX-STANDARD	:
SCHNEIDER CPC464	:
SCHNEIDER CPC664,CPC6128	:
SINCLAIR ZX81	:
SINCLAIR SPECTRUM	:

GETKEY

Befehlsart: Anweisung

GETKEY **Variable**

Beschreibung: Der Befehl veranlaßt den Computer, mit der Ausführung so lange zu warten, bis man eine Taste drückt. Dann wird das entsprechende Zeichen der **Variablen** zugewiesen.

Theoretisch lassen sich dabei Zahlenvariablen ebenso wie Stringvariablen verwenden. In der Praxis sind jedoch nur Stringvariablen sinnvoll, denn sonst erfolgt bei Drücken einer Buchstaben-Taste eine Fehlermeldung.

Man kann auch mehrere Variablen verwenden. Sie werden dann mit Komma voneinander getrennt.

Beispiele:

```
10 GETKEYa$
20 PRINT a$
30 GOTO 10
```

```
10 GETKEYA$,B$,C$
```

Anmerkung zur Kompatibilität: Bei Rechnern, die über diesen Befehl nicht verfügen, kann man ihn ersetzen:
a) Rechner, die den Befehl INKEY (siehe INKEY) aufweisen:

```
10 a$=INKEY$:IF a$="" THEN 10
```

b) Rechner, die den Befehl GET (siehe GET) aufweisen:

```
10 GETa$:IF a$="" THEN 10
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
› COMMODORE 16,116,PLUS4    :  GETKEY Stringvariable,Stringvariable,...
› COMMODORE 128             :  GETKEY Stringvariable,Stringvariable,...
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

GO64

Befehlsart: Kommando

GO64

Beschreibung: Das Kommando verwenden wir, um beim Commodore 128 den 128er-Modus zu verlassen und in den C64-Modus zu gelangen. Der 128er-Modus verfügt über den Wortschatz BASIC 7.Ø und 128KByte-Speicher.

Der 64er-Modus ist voll kompatibel zum Commodore 64. Spiele und andere Programme, die für den C64 erworben wurden, sind nun lauffähig.

Nachdem man den Befehl GO64 eingegeben hat, fragt der Computer nach:

ARE YOU SURE?

Gibt man Y für Yes ein und drückt die RETURN-Taste, wird das im Speicher befindliche Programm gelöscht; man kommt in den 64er-Modus.

Wenn man N für No eingibt und die RETURN-Taste drückt, bleibt man im 128er-Modus, und der Befehl wird ignoriert.

Beispiel:

```
GO64
ARE YOU SURE? Y
```

Kurze Zeit später meldet sich der Computer mit einer Einschaltmeldung nach Art des Commodore 64.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  GO64
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

GOSUB...RETURN

Befehlsart: Anweisung

GOSUB **Zeilennummer** ...RETURN

Beschreibung: Mit GOSUB springt man in ein Unterprogramm, das mit der Programmzeile **Zeilennummer** beginnt und mit RETURN abgeschlossen wird. Durch RETURN springt das Programm zu der Anweisung, die auf den GOSUB-Befehl folgt, egal ob diese in derselben oder in einer neuen Zeile festgelegt ist.

Ein Unterprogramm ("Subroutine") läßt sich in einem Programm mehrmals anspringen, so daß man im Programm wertvollen Platz spart und wiederkehrende Programmabläufe nicht stets neu eingegeben werden müssen.

Beispiel:

```
100 PRINT"WELCHE RECHENART"
110 PRINT "ADDIEREN (A)"
120 PRINT "SUBTRAHIEREN (S)"
130 INPUT X$
140 IF X$="A" THEN GOTO 180
150 IF X$="S" THEN GOTO 200
160 PRINT "FALSCHE EINGABE"
170 GOTO 130
180 PRINT "ICH ADDIERE IHRE BEIDEN ZAHLEN":GOSUB 800
190 PRINT A;" + ";B;" = ";A+B:GOTO 100
200 PRINT "ICH SUBTRAHIERE IHRE BEIDEN ZAHLEN":GOSUB 800
210 PRINT A;" - ";B;" = ";A-B:GOTO 100
800 INPUT "ZAHL 1";A
810 INPUT "ZAHL 2";B
820 RETURN
```

Da sich in diesem Programm zwar die Rechenarten ändern können, nicht aber die Eingabe der beiden Zahlen, kann man die Eingabe in einem Unterprogramm bearbeiten.

Wird ein RETURN ohne GOSUB verwendet, kommt es zu einer Fehlermeldung. Dasselbe geschieht, wenn das Programm die Subroutine erreichen kann und sie nun sozusagen als "nor-

male" Programmzeilen abarbeiten will. Dies würde etwa geschehen, falls in unserem Beispiel in den Zeilen 19Ø und 21Ø der Sprungbefehl GOTO 1ØØ nicht vorhanden wäre.

Anmerkung: In einem Unterprogramm läßt sich auch ein weiteres Unterprogramm aufrufen, und in diesem kann man wiederum ein anderes Unterprogramm aufrufen. Die Anzahl der erlaubten Verschachtelungen ist allerdings bei den verschiedenen Computertypen unterschiedlich.

Beispiel:

```
1ØØ
11Ø GOSUB 1ØØØ
12Ø

1ØØØ REM --- UNTERPROGRAMM 1 ---
1Ø1Ø
1Ø2Ø
1Ø9Ø GOSUB 2ØØØ
123Ø RETURN

2ØØØ REM -- UNTERPROGRAMM 2 --
2Ø1Ø
2Ø2Ø
24ØØ RETURN
```

Das zweite RETURN (Zeile 24ØØ) bringt das Programm auf Zeile 11ØØ, also auf die Anweisung direkt nach dem Sprungbefehl, zurück. Danach wird das Unterprogramm 1 weiter abgearbeitet, bis Zeile 123Ø erreicht ist. Sie bewirkt einen Rücksprung zu Zeile 12Ø und schließt damit das Unterprogramm ab.

Problematisch ist die Benutzung einer GOTO-Anweisung in Unterprogrammen. Sie darf nur verwendet werden, um Zeilen innerhalb des Unterprogramms anzuspringen.

Beispiel:

Richtig ist

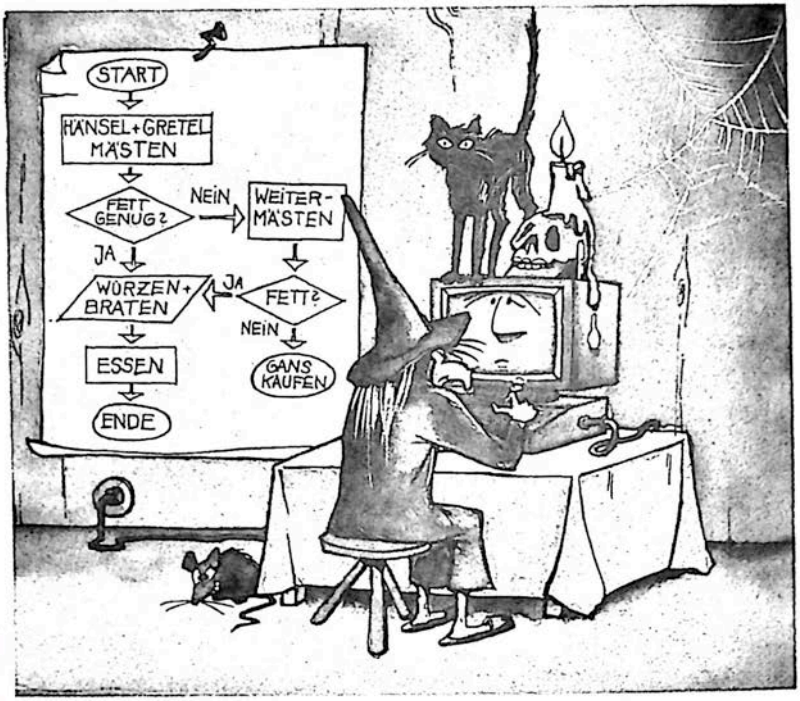
```
1ØØ
11Ø GOSUB 1ØØØ
12Ø

1ØØØ REM ---UNTERPROGRAMM---
1Ø1Ø PRINT F/H:K=K+1
1Ø2Ø IF K  2Ø THEN GOTO 1Ø1Ø
1Ø3Ø RETURN
```

Falsch ist

```
1ØØ
11Ø GOSUB 1ØØØ
12Ø

1ØØØ REM ---UNTERPROGRAMM---
1Ø1Ø PRINT F/H:K=K+1
1Ø2Ø IF K  2Ø THEN GOTO 15Ø
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	GOSUB Zeilennummer
› COMMODORE VC20,64	:	GOSUB Zeilennummer
› COMMODORE 16,116,PLUS4	:	GOSUB Zeilennummer
› COMMODORE 128	:	GOSUB Zeilennummer
› ENTERPRISE 64,128	:	GOSUB Zeilennummer
› MSX-STANDARD	:	GOSUB Zeilennummer
› SCHNEIDER CPC464	:	GOSUB Zeilennummer
› SCHNEIDER CPC664,CPC6128	:	GOSUB Zeilennummer
› SINCLAIR ZX81	:	GOSUB Zeilennummer
› SINCLAIR SPECTRUM	:	GOSUB Zeilennummer



GOTO

Befehlsart: Anweisung

GOTO **Zeilennummer**
GO TO **Zeilennummer**

Beschreibung: Durch den "unbedingten Sprungbefehl" GOTO springt das Programm zur angegebenen **Zeilennummer.** Wenn die **Zeilennummer** eine Programmzeile mit ausführbaren Anweisungen bezeichnet, werden diese Programmzeile und die darauffolgenden Programmzeilen abgearbeitet.

Ist keine Programmzeile mit der **Zeilennummer** vorhanden, ergibt sich (fast) immer eine Fehlermeldung wie diese:

Beispiel:

```
110 PRINT"--------------------"
120 GOTO 125
130 PRINT"--------------------"
RUN
--------------------

?UNDEF'D STATEMENT  ERROR IN 120
READY.
```

Führt der Sprungbefehl zu einer Zeile, die keine ausführbare Anweisung enthält (zum Beispiel REM; siehe dort!), wird die darauffolgende Programmzeile abgearbeitet.

Beispiel:

```
10 PRINT "L"
20 GOSUB 150
30 PRINT "30":GOSUB 150:GOTO 120
40 PRINT "40":GOSUB 150:GOTO 100
50 PRINT "50":GOSUB 150:GOTO 30
60 PRINT "60":GOSUB 150:GOTO 120
70 PRINT "70":GOSUB 150:GOTO 90
80 PRINT "80":GOSUB 150:GOTO 50
90 PRINT "90":GOSUB 150:GOTO 90
100 PRINT "100":GOSUB 150: GOTO 70
110 PRINT "110":GOSUB 150:GOTO 60
120 PRINT "120":GOSUB 150:GOTO 40
130 PRINT "130":GOSUB 150:GOTO 110
150 FOR I=1 TO 500: NEXT: PRINT "LLLLLLLLWIR SIND IM PROGRAMMZEILE:";
160 RETURN

   WIR SIND IN PROGRAMMZEILE:120
BREAK IN 150
READY.
GOTO 150
```

Anmerkung:

Mit

GOTO **Zeilennummer**

im Direktmodus eingegeben, kann man ein Programm starten, das durch BREAK, STOP etc. unterbrochen wurde. Das Programm beginnt dann mit der Abarbeitung in jener Programmzeile, die durch die **Zeilennummer** gekennzeichnet wird. Das ist (im Gegensatz zu CONT; siehe dort!) auch dann möglich,

wenn zwischen dem Abbruch und dem erneuten Start des Programmes Veränderungen vorgenommen worden sind.

Anmerkung zur Kompatibilität: Bei einigen wenigen Computern sind als Zeilennummern auch Rechenausdrücke erlaubt. "Die Möglichkeit, Sprungziele zu berechnen, die mit GOTO angesprungen werden sollen, findet man verhältnismäßig selten. Die Verwendung von Labels als Sprungziel wird in einem Normvorschlag für ein erweitertes BASIC gefordert, ist aber heute noch selten verwirklicht" (Dr. Hans-Joachim Sacht, "Programmiersprache BASIC - Schritt für Schritt", 1983 im Humboldt-Taschenbuchverlag Jacobi KG, München).

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  GOTO Zeilennummer
> COMMODORE VC20,64         :  GOTO Zeilennummer
> COMMODORE 16,116,PLUS4    :  GOTO Zeilennummer
> COMMODORE 128             :  GOTO Zeilennummer
> ENTERPRISE 64,128         :  GOTO Zeilennummer
> MSX-STANDARD              :  GOTO Zeilennummer
> SCHNEIDER CPC464          :  GOTO Zeilennummer
> SCHNEIDER CPC664,CPC6128  :  GOTO Zeilennummer
> SINCLAIR ZX81             :  GOTO Zeilennummer
> SINCLAIR SPECTRUM         :  GOTO Zeilennummer
```

GRAPHIC

Befehlsart: Anweisung

GRAPHIC **Zahl 1,Zahl 2,Zahl 3**
GRAPHIC CLR

Beschreibung: Dieser Befehl schaltet den Grafik-Modus ein und reserviert damit einen bestimmten Speicherbereich für die Grafiken. Mit

GRAPHIC CLR

wird dieser Modus wieder verlassen und der Speicherbereich wieder für Programme freigegeben.

Die **Zahl 1** gibt die Grafikbetriebsart an. Es gibt insgesamt 6 Grafikbetriebsarten, die mit den Zahlenwerten Ø bis 5 gekennzeichnet werden. Es bedeutet für

Zahl 1

Ø	Text mit 4Ø Zeichen pro Zeile
1	hochauflösende Grafik mit 32Ø 2ØØ Pixeln
2	hochauflösende Grafik mit Teil-Bildschirm für Text
3	Mehrfarbengrafik mit 16Ø 2ØØ Pixeln
4	Mehrfarbengrafik mit Teil-Bildschirm für Text
5	Text mit 8Ø Zeichen pro Zeile

Die **Zahl 2** gibt an, ob bei den Grafik-Modi (**Zahl 1** = 1 bis 4) mit dem Aufrufen gelöscht werden soll. **Zahl 2** kann den Wert Ø (nicht löschen) oder 1 (löschen) annehmen.

Die **Zahl 3** gibt an, ab welcher Zeile der Textbildschirm im Modus 2 oder im Modus 4 beginnen soll. Wenn man für sie keinen Wert angibt, wird automatisch Zeile 19 angewählt, d. h. die 2Ø. Zeile auf dem Bildschirm. Der **Zahl 3** lassen sich Werte zwischen Ø und 24 zuordnen.

Beispiele:

GRAPHIC 5	schaltet auf den 8Ø-Zeichen-Betrieb
GRAPHIC 1,1	schaltet auf den hochauflösenden Grafik-Modus um; der Bildschirm wird vorher gelöscht
GRAPHIC 4,Ø,12	schaltet auf den Mehrfarben-Modus um; der Bildschirm wird nicht gelöscht. Es wird ein Textbildschirm zur Verfügung gestellt, der mit Zeile 12 (13. Bildschirmzeile) beginnt.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	GRAPHICS Zahl1
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	GRAPHIC Zahl1,Zahl2,Zahl3
› COMMODORE 128	:	GRAPHIC Zahl1,Zahl2,Zahl3
ENTERPRISE 64,128	:	GRAPHICS HIRES Zahl GRAPHICS LORES Zahl
MSX-STANDARD	:	SCREEN Zahl1
SCHNEIDER CPC464	:	MODE Zahl1
SCHNEIDER CPC664,CPC6128	:	MODE Zahl1
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

GRAPHICS

Befehlsart: Anweisung

GRAPHICS **Zahl**

Beschreibung: Mit diesem Befehl wählt man eine der 16 Grafikbetriebsarten aus. Die ersten drei Betriebsarten stehen für Text zur Verfügung, die restlichen 13 für die Grafik-Darstellung mit unterschiedlicher Auflösung.

Die Betriebsart wird durch die **Zahl** angegeben. Im Programmiermodus läßt sich das Textfenster ausblenden, indem man zum jeweiligen Wert für die Grafikbetriebsart die Zahl 16 addiert. Addieren wir 32, wird der Bildschirm nicht gelöscht. Beide Möglichkeiten lassen sich auch miteinander verbinden.

Beispiel:

```
10 GRAPHICS 4+16
20 COLOR 1
30 PLOT 0,0
40 DRAWTO 79,47
50 GOTO 10
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	: GRAPHICS Zahl1
COMMODORE VC20,64	:
COMMODORE 16,116,PLUS4	: GRAPHIC Zahl1,Zahl2,Zahl3
COMMODORE 128	: GRAPHIC Zahl1,Zahl2,Zahl3
ENTERPRISE 64,128	: GRAPHICS HIRES Zahl GRAPHICS LORES Zahl
MSX-STANDARD	: SCREEN Zahl1
SCHNEIDER CPC464	: MODE Zahl1
SCHNEIDER CPC664,CPC6128	: MODE Zahl1
SINCLAIR ZX81	:
SINCLAIR SPECTRUM	:

GRAPHICS

Befehlsart: Anweisung

GRAPHICS
GRAPHICS HIRES **Zahl**
GRAPHICS LORES **Zahl**

Beschreibung: Mit dieser Anweisung wird der Grafik-Bildschirm eingeschaltet (siehe auch GRAPHIC). Die unteren vier Zeilen bleiben dem Anwender zur Textdarstellung.
Durch den Befehl

GRAPHICS

wird von der Textseite auf die Grafikseite umgeschaltet. Die Parameter für die Grafikseite werden von der zuletzt eingestellten Grafikseite übernommen.
Durch den Befehl

GRAPHICS HIRES **Zahl**

wird die hochauflösende Grafikseite eingeschaltet. Die **Zahl** gibt dabei die Anzahl der Farben wieder, die verwendet werden sollen. Zulässig sind die Werte 2, 4, 16, 256. Wird kein Wert angegeben, nimmt der Rechner automatisch die Farbenzahl der zuletzt verwendeten Grafikseite an.

Durch den Befehl

GRAPHICS LORES **Zahl**

wird die niedrigauflösende Grafikseite eingeschaltet. Anhand der **Zahl** läßt sich wiederum die Anzahl der verwendeten Farben bestimmen.

Beispiel:

```
10 GRAPHICS HIRES 256
20 X=INT(RND*1024)
30 Y=INT(RND*624)
40 SET INK INT(RND*256)
50 SET BEAM ON
60 PLOT X,Y
70 GOTO 20
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	GRAPHICS Zahl1
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	GRAPHIC Zahl1,Zahl2,Zahl3
COMMODORE 128	:	GRAPHIC Zahl1,Zahl2,Zahl3
› ENTERPRISE 64,128	:	GRAPHICS HIRES Zahl GRAPHICS LORES Zahl
MSX-STANDARD	:	SCREEN Zahl1
SCHNEIDER CPC464	:	MODE Zahl1
SCHNEIDER CPC664,CPC6128	:	MODE Zahl1
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

GSHAPE

Befehlsart: Anweisung

GSHAPE **String,Zahl 1,Zahl 2,Zahl 3**

Beschreibung: Anhand dieser Anweisung kann man ein "Shape", eine Figur, auf dem Grafikbildschirm abbilden.
Der **String** enthält die Bildinformationen für die Figur. Sie wurden ihm durch den Befehl SSHAPE (siehe dort) übergeben.

Zahl 1 gibt die Abszisse (Wert auf der x-Achse) für den oberen rechten Randpunkt der Figur an; **Zahl 2** den Wert auf der y-Achse (Ordinate). **Zahl 1** kann Werte zwischen Ø und 32Ø annehmen, **Zahl 2** Werte zwischen Ø und 2ØØ.

Die **Zahl 3** gibt an, wie die Figur gegenüber dem vorhandenen Hintergrund dargestellt werden soll.

Dabei bedeutet

Zahl 3

Ø Die Figur wird ohne Veränderung der Daten ausgegeben.

1 Die Figur wird invertiert ausgegeben.

2 Die Figur wird mit dem Hintergrund ODER-verknüpft (siehe auch unter OR).

3 Die Figur wird mit dem Hintergrund UND-verknüpft (siehe auch unter AND).

4 Die Punkte der Figur, die sich mit Punkten des Hintergrundes überschneiden, werden invertiert.

Beispiel:

```
10 GRAPHIC1,1
20 FOR I=1 TO 75
30 CIRCLE 1,100,100,I*4
40 NEXT I
50 X=INT(RND(1)*32)*10
60 Y=INT(RND(1)*20)*10
70 SSHAPE A$,X,Y,X+9,Y+9
80 X=INT(RND(1)*32)*10
90 Y=INT(RND(1)*20)*10
100 GSHAPE A$,X,Y,0
110 GOTO 50
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	GSHAPE Variable,Zahl1,Zahl2,Zahl3
› COMMODORE 128	:	GSHAPE Variable,Zahl1,Zahl2,Zahl3
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

HANDLER

Befehlsart: Anweisung

HANDLER **String**

END HANDLER

Beschreibung: Mit der Anweisung HANDLER kann ein Unterprogramm definiert werden, in das das Programm verzweigt, wenn ein Fehler oder ein CAUSE EXCEPTION (siehe dort) auftritt und mit WHEN EXCEPTION der Sprung dorthin definiert wurde.

Der **String** gibt den Namen der Unterroutine an.

Genauere Beschreibung zum Einsatz des Befehls HANDLER ist unter WHEN EXCEPTION zu finden.

Beispiel:

```
10 WHEN EXCEPTION USE FALSCH
20 PRINT "WELCHE ZAHL SOLL ICH MAL 5 NEHMEN"
30 INPUT A
40 PRINT A;" * 5 = ";
50 INPUT B
60 IF B<>A*5 THEN CAUSE EXCEPTION
70 END WHEN
80 END
500 HANDLER FALSCH
510    IF B>A*5 THEN PRINT "ZU GROSS!"
520    IF B<A*5 THEN PRINT "ZU KLEIN!"
530    RETRY
540 END HANDLER
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	siehe Beschreibung
MSX STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

HEADER

Befehlsart: Kommando

HEADER "**Disketten-Name**",I**id**,D**Zahl 1** ON U**Zahl 2**
HEADER "**Disketten-Name**",I**id**,D**Zahl 1**,U**Zahl 2**

Beschreibung: Mit dem Kommando HEADER formatieren wir eine Diskette. Auf diese Weise kann man auch eine Diskette löschen.

Der **Disketten-Name**, unter dem die Diskette formatiert wird, darf bis zu 16 Zeichen lang sein. Die **id**-Zahl, mit der man eine Diskette zum Beispiel numerieren kann, ist bis zu zwei Zeichen lang. **Zahl 1** gibt die Nummer des Laufwerkes an, in dem die Diskette formatiert werden soll. Diese Angabe benötigen wir nur, falls wir mit zwei Einzellaufwerken arbeiten. Möglich sind dabei die Werte 8,9,1Ø.

Beispiele:

HEADER "SPIELE",IØ1

Eine Diskette wird formatiert und erhält den Namen SPIELE und die id Ø1.

HEADER "TEXTVERARBEITUNG",IØ5,D1

Die Diskette in Laufwerk 1 wird formatiert. Sie erhält den Namen TEXTVERARBEITUNG und die id Ø5.

Anmerkung zur Kompatibilität: Beim C64 muß man das HEADER-Kommando in folgendes Format umschreiben:

OPEN1,**Zahl 2**,15,":**Disketten-Name**,**id**":CLOSE1

Die **Zahl 2** hat dabei im allgemeinen den Wert 8.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	OPEN1,Zahl2,15,"N:Disknane,Id"
COMMODORE 16,116,PLUS4	:	HEADER "Diskname",IId,DZahl1,UZahl2
COMMODORE 128	:	HEADER "Diskname",IId,DZahl1,UZahl2
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	Disk wird mit C/PM-Befehl formatiert
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	FORMAT "M":Zahl1:"Diskname"

HELP

Befehlsart: Kommando

HELP

Beschreibung: Dieses Kommando hilft dem Benutzer bei der Fehlersuche. Sobald in einer Zeile ein Programmierfehler auftritt und die entsprechende Fehlermeldung ausgegeben worden ist, bewirkt das Kommando HELP, daß die fehlerhafte Zeile auf dem Bildschirm gezeigt wird. Die Zeile blinkt dabei ab dem durch den Computer entdeckten Fehler und weist damit auf den zu verbessernden Teil hin.

Beispiel:

```
1Ø IF i=Ø THEN PRINT:PRNT:PRINT
RUN

? SYNTAX ERROR IN 1Ø
READY.
HELP

1Ø IF i=Ø THEN PRINT:PRNT:PRINT
                     ← Ab hier blinkt die Zeile.
```

Kompatibilitätsliste:

```
ATARI 600,800,130XE       :
COMMODORE VC20,64         :
COMMODORE 16,116,PLUS4    :  HELP
COMMODORE 128             :  HELP
ENTERPRISE 64,128         :
MSX-STANDARD              :
SCHNEIDER CPC464          :
SCHNEIDER CPC664,CPC6128  :
SINCLAIR ZX81             :
SINCLAIR SPECTRUM         :
```

HEX$

Befehlsart: Funktion

HEX$(**Zahl 1,Zahl 2**)

Beschreibung: Die Dezimalzahl **Zahl 1** wird in eine hexadezimale Zahl umgewandelt.

Mit **Zahl 2** kann man die Anzahl der Stellen dieser hexadezimalen Zahl angeben. Sie darf auch weggelassen werden.

Das hexadezimale System ist auf 16 Zeichen (ausgedrückt in Ziffern und Buchstaben) aufgebaut, das dezimale auf 1Ø Ziffern und das binäre Zahlensystem auf 2 Ziffern.

Beispiel:

```
10 FOR i=1 TO 60
20 PRINT "Die ganze Zahl ";i
30 PRINT "ist die hexadezimale Zahl ";HEX$(i,8)
40 FOR t=1 TO 800:NEXT t
50 PRINT:PRINT:PRINT
60 NEXT i
```

```
Die ganze Zahl  8
ist die hexadezimale Zahl 00000008
Die ganze Zahl  9
ist die hexadezimale Zahl 00000009
Die ganze Zahl  10
ist die hexadezimale Zahl 0000000A
Die ganze Zahl  11
ist die hexadezimale Zahl 0000000B
```

```
Die ganze Zahl  12
ist die hexadezimale Zahl 0000000C
Die ganze Zahl  13
ist die hexadezimale Zahl 0000000D
Die ganze Zahl  14
ist die hexadezimale Zahl 0000000E
Die ganze Zahl  15
ist die hexadezimale Zahl 0000000F
Die ganze Zahl  16
ist die hexadezimale Zahl 00000010
Die ganze Zahl  17
ist die hexadezimale Zahl 00000011
Die ganze Zahl  18
ist die hexadezimale Zahl 00000012
Die ganze Zahl  19
ist die hexadezimale Zahl 00000013
Die ganze Zahl  20
ist die hexadezimale Zahl 00000014
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
* COMMODORE 16,116,PLUS4    :  HEX$(Zahl1)
* COMMODORE 128             :  HEX$(Zahl1)
  ENTERPRISE 64,128         :
* MSX-STANDARD              :  HEX$(Zahl1)
* SCHNEIDER CPC464          :  HEX$(Zahl1,Zahl2)
* SCHNEIDER CPC664,CPC6128  :  HEX$(Zahl1,Zahl2)
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

HIMEM

Befehlsart: Variable

HIMEM

Beschreibung: Mit dem MEMORY-Befehl (siehe MEMORY) läßt sich die Obergrenze im RAM-Speicher festlegen, die BASIC zur Verfügung stehen soll. Diesen Adressenausdruck enthält die Variable HIMEM.

Mit

? HIMEM
oder

PRINT HIMEM

kann man jederzeit diese Grenze abfragen.

Beispiel:

```
? HIMEM
 43903
READY
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :  HIMEM
> SCHNEIDER CPC664,CPC6128  :  HIMEM
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

IF...THEN...ELSE

Befehlsart: Anweisung

IF **Variablenname="Textausdruck"** THEN **Anweisung** ELSE **Anweisung**
IF **Variablenname=String** THEN **Anweisung** ELSE **Anweisung**
IF **Variablenname=Rechenausdruck** THEN **Anweisung** ELSE **Anweisung**

IF...THEN...ELSE

Beschreibung: Mit dem Befehl veranlassen wir das Programm, eine bestimmte **Anweisung** auszuführen, falls der **Variablenname** eine bestimmte Bedingung erfüllt. Die Bedingung kann beispielsweise lauten, daß sich dem **Variablennamen** ein bestimmter Textausdruck oder ein bestimmter String oder auch ein Rechenausdruck zuordnen läßt. Die Bedingung kann ebensogut bedeuten, daß der **Variablenname** größer oder gleich oder kleiner bzw. größer-gleich oder kleiner-gleich einer bestimmten Variablen ist.

Ist die Bedingung nicht erfüllt, so kann man bei den meisten BASIC-Dialekten mit ELSE **Anweisung** die **Anweisung**, die hinter dem ELSE steht, abarbeiten lassen.

Beispiel:

```
10 PRINT "EINE RECHNUNG"
20 INPUT "MIT ";A
30 B=INT(RND(1)*10)+2
40 PRINT B;" * ";A
50 INPUT C
60 IF C=B*A THEN GOSUB 1000
61 PRINT "▒NEUE RECHNUNG? GEBE >J< EIN."
62 INPUT E$
63 IF E$="J" THEN GOTO 10
64 END
70 PRINT "LEIDER FALSCH"
80 PRINT "NOCHMAL: ":GOTO 50
1000 FOR X=1 TO 15
1010 PRINT "▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒▒RICHTIG!"
1015 FOR T=1TO 250: NEXT T
1020 POKE 53280,X
1030 NEXT X
1040 RETURN
```

Anmerkung zur Kompatibilität: Einige BASIC-Dialekte verfügen nicht über den ELSE-Befehl. Er muß dann durch GOTO-Kombinationen simuliert werden.

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  IF Bedingung THEN Anweisung
› COMMODORE VC20,64         :  IF Bedingung THEN Anweisung
› COMMODORE 16,116,PLUS4    :  IF Bedingung THEN Anweisung ELSE Anweisung
› COMMODORE 128             :  IF Bedingung THEN Anweisung ELSE Anweisung
› ENTERPRISE 64,128         :  IF Bedingung THEN Anweisung ELSE Anweisung
› MSX-STANDARD              :  IF Bedingung THEN Anweisung ELSE Anweisung
› SCHNEIDER CPC464          :  IF Bedingung THEN Anweisung ELSE Anweisung
› SCHNEIDER CPC664,CPC6128  :  IF Bedingung THEN Anweisung ELSE Anweisung
› SINCLAIR ZX81             :  IF Bedingung THEN Anweisung
› SINCLAIR SPECTRUM         :  IF Bedingung THEN Anweisung
```

IMAGE

Befehlsart: Anweisung

IMAGE: **String**

Beschreibung: Diese Anweisung unterstützt den Befehl PRINT USING (siehe dort).

Die Möglichkeiten dieses Befehls sind so umfangreich, daß nur durch wiederholtes Probieren Erfolge erzielt werden können. Hier empfiehlt es sich, mit Hilfe des jeweiligen Benutzerhandbuchs gründlich zu experimentieren.

Beispiel:

```
IMAGE: "####.###"
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :  PUDEF String
  COMMODORE 128             :  PUDEF String
› ENTERPRISE 64,128         :  IMAGE:String
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

IN

Befehlsart: Funktion

IN(Zahl)

Beschreibung: Mit dieser Funktion kann der Benutzer ein Byte von jenem Ein-/Ausgang, der durch die **Zahl** gekennzeichnet ist, in den Rechner einlesen. Die **Zahl** kann Werte zwischen 0 und 255 annehmen.

Beispiel:

```
PRINT IN(2)
 255
```

Anmerkung zur Kompatibilität: Ein verwandter Befehl ist INP (siehe dort).

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  IN(Zahl)
  MSX-STANDARD              :  INP(Zahl)
  SCHNEIDER CPC464          :  INP(Zahl)
  SCHNEIDER CPC664,CPC6128  :  INP(Zahl)
  SINCLAIR ZX81             :
> SINCLAIR SPECTRUM         :  IN Zahl
```

INF

Befehlsart: Konstante

INF

Beschreibung: Diese Konstante enthält den Wert, den sich der Enterprise unter einer unendlich großen Zahl vorstellt.
Der Wert dieser "unendlich großen" Zahl ist 9.999999999E62.

Beispiel:

```
10 REM *Die unendliche Geschichte*
20 k=1
30 PRINT "Die unendliche Geschichte"
40 PRINT "geht weiter..."
50 PRINT "und weiter...":IF k<inf THEN GOTO 50
60 PRINT "Aber jetzt ist sie doch aus."
```

Anmerkung zur Kompatibilität: Die "Vorstellung" von einer unendlich großen Zahl ist von Computertyp zu Computertyp verschieden.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  INF
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

INFO

Befehlsart: Kommando

INFO

Beschreibung: Dem Anwender wird die Anzahl der freien Speicherplätze angezeigt, und er wird darüber informiert, wie sich der benutzte Speicherplatz auf die einzelnen Programme verteilt.
Diesen Befehl kann man nur im Direktmodus verwenden.

Beispiel:

INFO

Anmerkung zur Kompatibilität: Dieser Befehl des ENTERPRISE ist bei anderen Rechnern durch das Kommando FRE (siehe dort) ersetzt, das allerdings auch innerhalb eines Programms verwendet werden darf.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	FRE(Zahl)
COMMODORE VC20,64	:	FRE(Zahl)
COMMODORE 16,116,PLUS4	:	FRE(Zahl)
COMMODORE 128	:	FRE(Zahl)
› ENTERPRISE 64,128	:	INFO
MSX-STANDARD	:	FRE(Zahl)
SCHNEIDER CPC464	:	FRE(Zahl)
SCHNEIDER CPC664,CPC6128	:	FRE(Zahl)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

INK

Befehlsart: Anweisung

INK **Zahl 1,Zahl 2,Zahl 3**

Beschreibung: Mit diesem Befehl werden den sogenannten Farbstiften die entsprechenden Farben zugeordnet. Man stellt sich sozusagen einen "Farbkasten" zusammen, der verschiedene Farbstifte enthält. Nachdem man bestimmten Farbstiften ihre Farbe zugeordnet hat, lassen sie sich später im Programm bei Bedarf aufrufen.

Zahl 1 bezeichnet die Nummer des Farbstiftes. Mit dieser Nummer rufen wir den entsprechenden Farbstift auch im Programm wieder auf. Es stehen insgesamt 16 Farbstifte zur Verfügung; **Zahl 1** kann daher einen Wert zwischen Ø und 15 annehmen.

Zahl 2 bezeichnet die Farbe des Stiftes. Sie kann zwischen Ø (schwarz) und 26 (leuchtendweiß) liegen.

Mit **Zahl 3** kann man eine zweite Farbe angeben, deren Wert ebenfalls zwischen Ø und 26 liegen muß. In diesem Falle blinkt der so definierte Farbstift zwischen den beiden angegebenen Farbwerten.

Die Blinkfrequenz läßt sich hierbei mit Hilfe des Befehls SPEED INK (siehe SPEED INK) verändern.

Beispiel:

```
10 CLS
20 INK 2,13:REM farbstift 2, weiss
30 INK 3,0,9:REM farbstift 3, blinkend schwarz-gruen
40 PRINT "Farbtest"
50 PAPER 2
60 PRINT "Hintergrund der Zeichen ist nun weiss (Druecke ENTER-Taste)."
70 INPUT a$
80 PEN 3
90 PRINT "Die Farbe der Zeichen blinkt schwarz-gruen."
100 END
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :  INK Zahl1,Zahl2,Zahl3
> SCHNEIDER CPC664,CPC6128  :  INK Zahl1,Zahl2,Zahl3
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

INKEY

Befehlsart: Funktion

INKEY(**Zahl**)

Beschreibung: Mit dieser Funktion kann man feststellen, ob die Taste **Zahl** gedrückt wurde. Dabei gibt es folgende Wertzuweisungen:

WERT	SHIFT	CTRL	TASTE
-1	?	?	nicht gedrückt
Ø	nicht gedrückt	nicht gedrückt	gedrückt
32	gedrückt	nicht gedrückt	gedrückt
128	nicht gedrückt	gedrückt	gedrückt
16Ø	gedrückt	gedrückt	gedrückt

Die Nummernzuordnungen sind dem Handbuch des Schneider CPC 464, ANHANG III (bei Schneider CPC 6128, Kapitel 7) zu entnehmen.

Beispiel:

```
10 IF INKEY(55)=32 THEN PRINT "Es wurde V & SHIFT gedrueckt"ELSE 10
20 END
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	INKEY(Zahl)
› SCHNEIDER CPC664,CPC6128	:	INKEY(Zahl)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

INKEY$

Befehlsart: Funktion

INKEY$

Beschreibung: Ein Computer hat einen Tastaturpuffer. Je nach Computertyp speichert er einige Zeichen in diesem, bis diese vom Computer abgerufen werden, um diese auf dem Bildschirm auszugeben.

Mit dem Befehl INKEY$ wird ein Zeichen aus dem Tastaturpuffer geholt. Man kann es dann einer Stringvariablen zuweisen. Falls kein Zeichen mehr im Tastaturpuffer steht, erhält man einen Leerstring.

Dieser Befehl ist ähnlich dem GET-Befehl (siehe dort).

Beispiel:

```
10 PRINT "Bin ich ein kluger Computer?"
20 PRINT "  (j=ja/n=nein)"
30 PRINT
40 a$=INKEY$
50 IF a$="" THEN 40
60 IF a$="j" THEN PRINT "Danke fuer das
                                    Komplement!":END
70 IF a$="n" THEN PRINT "Dies will ich nicht
80 GOTO 40                          hoeren!":PRINT:RUN
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	GET Stringvariable
COMMODORE 16,116,PLUS4	:	GET Stringvariable
COMMODORE 128	:	GET Stringvariable
> ENTERPRISE 64,128	:	Stringvariable=INKEY$
> MSX-STANDARD	:	Stringvariable=INKEY$
> SCHNEIDER CPC464	:	Stringvariable=INKEY$
> SCHNEIDER CPC664,CPC6128	:	Stringvariable=INKEY$
> SINCLAIR ZX81	:	LET Stringvariable=INKEY$
> SINCLAIR SPECTRUM	:	LET Stringvariable=INKEY$

INP

Befehlsart: Funktion

INP(**Schnittstellenadresse**)

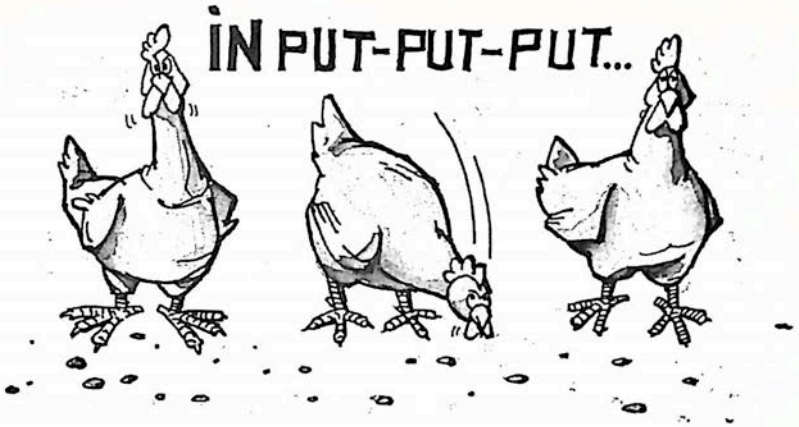
Beschreibung: Mit INP kann man den Userport des Rechners danach abfragen, ob eine Information vorhanden ist.

Beispiel:

```
? INP(&FD11)
 Ø
READY
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	IN(Schnittstellenadresse)
> MSX-STANDARD	:	INP(Schnittstellenadresse)
> SCHNEIDER CPC464	:	INP(Schnittstellenadresse)
> SCHNEIDER CPC664,CPC6128	:	INP(Schnittstellenadresse)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	IN Zahl



INPUT

Befehlsart: Anweisung

INPUT **Variablenliste**
INPUT **Zahl;"Ausgabetext";Variablenliste**

Beschreibung: Die INPUT-Anweisung ermöglicht dem Benutzer Eingaben von Daten während der Bearbeitung eines Programmes. Sobald das Programm die INPUT-Anweisung erreicht, hält es mit der Ausführung inne und verlangt mit einem Fragezeichen (?) nach einer Eingabe. Die Eingabe muß mit der RETURN- bzw. ENTER-Taste dem Rechner übergeben werden.

In der Variablenliste lassen sich sowohl numerische Variablen als auch Stringvariablen abfragen. Die einzelnen Elemente in der **Variablenliste** sind durch Kommata voneinander getrennt.

Beispiel:

```
10 INPUT A
20 INPUT B
30 PRINT "DIE SUMME IHRER BEIDEN EINGEGEBENEN  ZAHLEN: ";A+B

READY.
```

Im zweiten Format des Befehls gibt die **Zahl** das Ein- oder Ausgabegerät an.

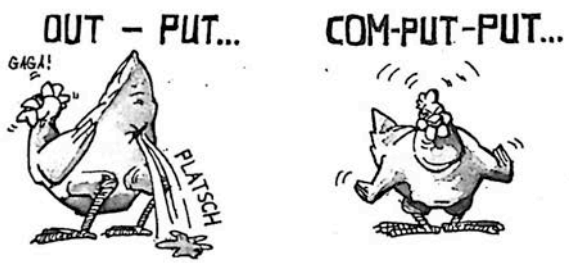
In einem **Ausgabetext** kann man zusätzliche Informationen unterbringen. Er wird im Format des Befehls, vergleichbar einer eingeschobenen PRINT-Anweisung (siehe dort), bearbeitet.

Beispiele:

```
10 INPUT "WIE HEISSEN SIE";A$
20 INPUT "UND WIE ALT SIND SIE(BITTE NUR ZAHL)";A
30 PRINT A$;" IST ";A;" JAHRE ALT."

READY.

10 PRINT "GEBEN SIE EINE REIHE VON ZAHLEN EIN."
20 PRINT "WIEVIEL ZAHLEN (1-20) WOLLEN SIE EINGE- BEN"
30 INPUT A
40 IF A>20 OR A<1 THEN PRINT "GILT NICHT!":GOTO 20
50 FOR B =1 TO A
60 PRINT"IHRE ZAHL: ":INPUT C:D=D+C
70 NEXT B
80 PRINT "DIE QUERSUMME IHRER ZAHLEN IST: ";D/A

READY.
```

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  INPUT Variablenliste
› COMMODORE VC20,64         :  INPUT TEXT;Variablenliste
› COMMODORE 16,116,PLUS4    :  INPUT TEXT;Variablenliste
  COMMODORE 128             :  INPUT TEXT;Variablenliste
› ENTERPRISE 64,128         :  INPUT PROMPT TEXT:Variablenliste
› MSX-STANDARD              :  INPUT TEXT;Variablenliste
› SCHNEIDER CPC464          :  INPUT TEXT;Variablenliste
› SCHNEIDER CPC664,CPC6128  :  INPUT TEXT;Variablenliste
› SINCLAIR ZX81             :  INPUT Variablenliste
› SINCLAIR SPECTRUM         :  INPUT TEXT;Variablenliste
```



INPUT$

Befehlsart: Funktion

INPUT$ (**Zahl, Dateinummer**)

Beschreibung: Die Funktion bestimmt die Länge eines Strings. Die **Zahl** gibt die Zeichen-Anzahl des Strings wieder. Die Eingabe kann über die Tastatur erfolgen oder, wenn die **Dateinummer** angegeben ist, von einem Speichermedium übernommen werden.

Beispiel:

```
500 open "LEUTE" FOR INPUT AS #1
510 INPUT #1, NA$,VN$
520 PRINT NA$,VN$
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  INPUT#(Zahl,#Dateinummer)
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

INPUT

Befehlsart: Anweisung

INPUT #**Dateinummer;Variablenliste**

Beschreibung: Der Befehl wirkt ähnlich der INPUT-Anweisung (siehe dort). Allerdings werden Daten nicht über die Ta-

statur eingegeben, sondern aus einer mit der **Dateinummer** angegebenen Datei gelesen.

Beispiel:
```
1000 PRINT "DATEN IN 12 ZEICHEN"
1010 F$=INPUT $(6)
```

Anmerkungen zur Kompatibilität:

Besonderheit

Im ATARI-BASIC kann man anhand des Formats

INPUT #Zahl:Stringvariable

bewirken, daß Daten von jenem Kanal der **Stringvariablen** übertragen werden, der durch die **Zahl** gekennzeichnet ist.

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  INPUT #Dateinr,Variablenliste
> COMMODORE VC20,64         :  INPUT #Dateinr,Variablenliste
> COMMODORE 16,116,PLUS4    :  INPUT #Dateinr,Variablenliste
> COMMODORE 128             :  INPUT #Dateinr,Variablenliste
> ENTERPRISE 64,128         :
                         INPUT #Dateinr,PROMPT TEXT:Variablenliste
> MSX-STANDARD              :  INPUT #Dateinr,Variablenliste
> SCHNEIDER CPC464          :  INPUT #Dateinr,TEXT;Variablenliste
> SCHNEIDER CPC664,CPC6128  :  INPUT #Dateinr,TEXT;Variablenliste
  SINCLAIR ZX81             :
> SINCLAIR SPECTRUM         :  INPUT #Dateinr;TEXT;Variablenliste
```

INSTR

Befehlsart: Funktion

INSTR(**Zahl 1,String,Teilstring**)

Beschreibung: Mit dem Befehl INSTR kann man innerhalb des **Strings** nach einem **Teilstring** suchen lassen. Die Funktion

gibt dann die erste Position jener Stelle an, an der der Teilstring gefunden wurde. Die Suche beginnt am Anfang des String-Textes oder an der Stelle, die durch die **Zahl** angegeben wird.

Wird der **Teilstring** im **String** nicht gefunden, wird der Wert Ø zurückgegeben.

Mit dieser Funktion kann man in einem Text ein bestimmtes Wort, einen Wortteil, einen Buchstaben usw. suchen.

Beispiel:

```
10 INPUT "Gib einen Text ein";a$
20 i=1
30 i=INSTR(i+1,a$,"e")
40 IF i>0 THEN e=e+1
50 IF i<>0 AND i+1<=LEN(a$) THEN 30
60 PRINT "In dem Text ist"e"mal der Buchstabe e enthalten."
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  INSTR(String,Teilstring,Zahl)
> COMMODORE 128             :  INSTR(String,Teilstring,Zahl)
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  INSTR(Zahl,String,Teilstring)
> SCHNEIDER CPC464          :  INSTR(Zahl,String,Teilstring)
> SCHNEIDER CPC664,CPC6128  :  INSTR(Zahl,String,Teilstring)
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

INT

Befehlsart: Funktion

INT(**Zahl**)

Beschreibung: Die Funktion rundet eine Bruch-Zahl auf die nächstkleinere Zahl ab. Man erhält die größte ganze Zahl, die kleiner (oder gleich) dieser **Zahl** ist. Die **Zahl** muß in Klammern hinter dem Befehl stehen.

INT verhält sich demnach bei negativen Zahlen anders als etwa die Funktion FIX (siehe FIX). Die Funktion FIX beseitigt bei jeder Zahl die Stellen hinter dem Komma.

Beispiele:

```
PRINT INT(3.85)
 3
READY.

PRINT INT(7.999)
 7
READY.
```

```
PRINT INT(-3.85)
-4
READY.

PRINT INT(2)
 2
READY.
```

Anmerkungen: Durch den Einsatz der INT-Funktion kann man sehr verschiedene Ergebnisse erzielen, je nach der Position im Programm, an der sie eingesetzt wird.

Nachstehend zwei Beispiele mit ähnlich erscheinenden Programmen, die sehr unterschiedliche Ergebnisse erzielen, weil die INT-Funktion an verschiedenen Stellen eingefügt wurde:

```
10 FOR I=1 TO 10 STEP 0.25
20 K=INT(SQR(I)):L=L+K:PRINT K;
30 NEXT I
40 PRINT L
```

```
10 FOR I=1.05 TO 3.00 STEP 0.25
20 FOR J=3.99 TO 4.09 STEP 0.125
30 PRINT INT (I) + INT(J);
40 NEXT J
50 NEXT I
RUN
 4  4  4  4  5  5  5  5  6
READY.
```

```
10 FOR I=1 TO 10 STEP 0.25
20 K=SQR(I):L=L+K:PRINT K;
30 NEXT I
40 PRINT INT(L)
```

```
10 FOR I=1.05 TO 3.00 STEP 0.25
20 FOR J=3.99 TO 4.09 STEP 0.125
30 PRINT INT (I+J);
40 NEXT J
50 NEXT I
RUN
 5  5  5  5  6  6  6  6  7
READY.
```

Je genauer der Wert einer Zahl sein soll, die mit Hilfe der INT-Funktion gerundet wird, desto später im Programm sollte man sie einsetzen. Das genauere Ergebnis erzielt man, wenn die INT-Funktion nicht innerhalb, sondern nach einer Schleife eingesetzt wird.

Anmerkung zur Kompatibilität: In manchen BASIC-Dialekten läßt sich die INT-Funktion verwenden, um eine ganzzahlige Zufallszahl zu erhalten.
Ein Beispiel für den Commodore 64:

```
10 FOR I=1TO10
20 PRINT INT(RND(1)*10)+1;
30 NEXT I
```

Das Programm erzeugt Zufallszahlen zwischen 1 und 1Ø.

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  INT(Zahl)
› COMMODORE VC20,64         :  INT(Zahl)
› COMMODORE 16,116,PLUS4    :  INT(Zahl)
› COMMODORE 128             :  INT(Zahl)
› ENTERPRISE 64,128         :  INT(Zahl)
› MSX-STANDARD              :  INT(Zahl)
› SCHNEIDER CPC464          :  INT(Zahl)
› SCHNEIDER CPC664,CPC6128  :  INT(Zahl)
› SINCLAIR ZX81             :  INT Zahl
› SINCLAIR SPECTRUM         :  INT Zahl
```

INTERVALL ON/OFF/STOP

Befehlsart: Anweisung

INTERVALL ON
INTERVALL OFF
INTERVALL STOP

Beschreibung: Dieser Befehl ist nur in Verbindung mit dem Befehl ON INTERVALL GOSUB (siehe dort!) sinnvoll.

INTERVALL ON	ermöglicht Unterbrechungen durch den ON...-Befehl (siehe ON)
INTERVALL OFF	schaltet den INTERVALL-Befehl ab. Mit INTERVALL ON wird er wieder eingeschaltet.
INTERVALL STOP	verhindert eine Unterbrechung in dem durch die Unterbrechung aufgerufenen Unterprogramm. Nach einem RETURN-Befehl (siehe RETURN) wird sofort in das Unterprogramm verzweigt.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
> MSX-STANDARD              :
            INTERVAL ON  /  INTERVAL OFF  /  INTERVALL STOP
  SCHNEIDER CPC464          :  EI  /  DI  /  -
  SCHNEIDER CPC664,CPC6128  :  EI  /  DI  /  -
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

INVERSE

Befehlsart: Anweisung

INVERSE **Zahl**

Beschreibung: Mit dieser Anweisung kann man Zeichen invers ausgeben lassen. Das heißt, falls die Anweisung gültig ist, wird dem Zeichen die Hintergrundfarbe zugewiesen und dem Zeichenhintergrund die Zeichenfarbe zugeordnet.

Die **Zahl** kann die Werte 1 oder Ø annehmen. Der Wert 1 bewirkt die inverse Zeichenausgabe; der Wert Ø schaltet auf die gewohnte Ausgabe zurück.

Beispiel:

```
10 INVERSE 1
20 PRINT "Schneider"
30 INVERSE 0
40 PRINT "Buch ";
50 GOTO 10
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	CHR$(18)
COMMODORE 16,116,PLUS4	:	CHR$(18)
COMMODORE 128	:	CHR$(18)
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	CHR$(24)
SCHNEIDER CPC664,CPC6128	:	CHR$(24)
SINCLAIR ZX81	:	
› SINCLAIR SPECTRUM	:	INVERSE

IP

Befehlsart: Funktion

IP(**Zahl**)

Beschreibung: Eine **Zahl** wird als ganzzahliger Wert ohne die Nachkommastellen zurückgegeben.

Beispiel:

```
PRINT IP(1.23)
 1
READY

PRINT IP(-23.45)
-23
READY
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	IP(Zahl)
MSX-STANDARD	:	FIX(Zahl)
SCHNEIDER CPC464	:	FIX(Zahl)
SCHNEIDER CPC664,CPC6128	:	FIX(Zahl)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

JOY

Befehlsart: Funktion

JOY(**Zahl**)

Beschreibung: Neben der Tastatur hat man bei den meisten Computern auch die Möglichkeit, mittels eines Joysticks Informationen zu übergeben. Mit dem JOY-Befehl lassen sich bis zu zwei Joysticks daraufhin abfragen, wie sie bewegt werden und ob der Feuerknopf gedrückt worden ist. Die **Zahl** nimmt dabei von Computertyp zu Computertyp unterschiedliche Werte an (siehe Anmerkungen zur Kompatibilität).

Beispiel:

```
PRINT JOY(1)
 Ø
READY
```

Anmerkungen zur Kompatibilität:

Die **Zahl** beträgt bei für	C16/C128	Enterprise	Schneider CPC
Joystick 1	1	1	Ø
Joystick 2	2	2	1
eingebauter Joystick	nicht vorhanden		nicht vorhanden

Die wiedergegebenen Werte für die Bewegung und das Drücken des Feuerknopfes:

C16/C128				Schneider CPC				Enterprise			
8	1	2	128	5	1	9	16	1Ø	8	9	16
7	Ø	3		4	Ø	8		2	Ø	1	
6	5	4		6	2	1Ø		6	4	5	

Kompatibilitätsliste:

	ATARI 600,800,130XE	: STICK(Zahl)
	COMMODORE VC20,64	:
›	COMMODORE 16,116,PLUS4	: JOY(Zahl)
›	COMMODORE 128	: JOY(Zahl)
›	ENTERPRISE 64,128	: JOY(Zahl)
	MSX-STANDARD	: STICK(Zahl)
›	SCHNEIDER CPC464	: JOY(Zahl)
›	SCHNEIDER CPC664,CPC6128	: JOY(Zahl)
	SINCLAIR ZX81	:
	SINCLAIR SPECTRUM	:

KEY

Befehlsart: Kommando

KEY **Zahl,"Befehl"**

Beschreibung: Durch dieses Kommando kann man die Funktionstasten eines Rechners mit bestimmten **Befehl**en belegen. Sie werden dann nach Drücken der entsprechenden Funktionstasten automatisch geschrieben und lassen sich mit ENTER, RETURN usw. starten.

Die **Zahl** gibt die Nummer der neu zu belegenden Funktionstaste an. Sie ist normalerweise identisch mit der Nummer der Funktionstaste. Die **Zahl** 3 belegt also zum Beispiel bei den Commodore-Rechnern die Funktionstaste F3.

Der **Befehl** wird in Anführungszeichen geschrieben. Soll der Befehl automatisch, also ohne Betätigung der ENTER- oder RETURN-Taste, ausgeführt werden, muß der Befehl folgendes Format erhalten:

KEY **Zahl,"Befehl"**

Beispiel:

```
KEY 1,"RUN" CHR$(13)
```

KEY

Durch dieses Kommando wird die Funktionstaste 1 mit dem Befehl RUN belegt. Beim Drücken der Taste wird er automatisch ausgeführt.

Anmerkungen zur Kompatibilität: Bei den Commodore-Rechnern, die diesen Befehl besitzen, werden durch dieses Kommando ohne Angabe von **Zahl** und **Befehl** die Funktionstastenbelegungen aufgelistet.

Bei den Schneider-CPC-Computern wird durch das Kommando der Zehnerblock mit Befehlen belegt:

Taste	Ø	1	2	3	4	5	6	7	8	9	.	ENTER
Zahl	128	129	13Ø	131	132	133	134	135	136	137	138	139

Die Länge aller Anweisungen darf den Umfang von 128 Zeichen nicht übersteigen.

Bei den Enterprise-Rechnern heißt dieser Befehl FKEY (siehe dort).

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  KEY Zahl,Befehl
> COMMODORE 128             :  KEY Zahl,Befehl
  ENTERPRISE 64,128         :  SET FKEY Zahl Befehl
> MSX-STANDARD              :  KEY Zahl,Befehl
> SCHNEIDER CPC464          :  KEY Zahl,Befehl
> SCHNEIDER CPC664,CPC6128  :  KEY Zahl,Befehl
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

KEY DEF

Befehlsart: Kommando

KEY DEF **Tastennummer,Zahl,Zeichen 1,Zeichen 2,Zeichen 3**

Beschreibung: Mit diesem Kommando ist der Benutzer in der Lage, die Tasten der Schneider-Computer neu zu belegen. Es ist unter anderem möglich, anstelle der englischen eine deutsche Tastatur zu definieren. Daraufhin wird zum Beispiel durch Drücken der Y-Taste ein Z wiedergegeben.

Die **Tastennummer** gibt die Nummer der Taste wieder, die geändert werden soll. Die Nummern sind dem Handbuch des CPC 464, im Anhang III unter dem Stichwort "Nummernzuordnung zu Tasten und Joystick-Signalen" zu entnehmen.

Die **Zahl** gibt an, ob den jeweils neu definierten Tasten eine Wiederholungsfunktion (REPEAT) zugeordnet wird oder nicht. Durch 1 wird die Wiederholungs- oder Dauerfunktion erreicht, durch Ø wird sie aufgehoben.

Zeichen 1 muß nicht angegeben werden, wenn man nur die Wiederholfunktion dieser Taste aus- oder einschalten will. Gibt man eine Zahl zwischen Ø und 255 ein, so wird die Taste neu definiert, und zwar wird der Taste das entsprechende Zeichen des ASCII-Code (siehe unter ASC) zugeordnet.

Zeichen 2 muß man nur dann angeben, wenn eine Belegungsänderung der Taste mit SHIFT durchgeführt werden soll oder wenn ein **Zeichen 3** angegeben wird.

Zeichen 3 muß man nur dann angeben, wenn eine Belegungsänderung der Taste mit CTRL durchgeführt werden soll.

Beispiel:

```
10 FOR i=0 TO 15
20 KEY DEF i,1,INT(RND(1)*256),INT(RND(1)*256)
30 NEXT i
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :
          KEY DEF Tastennr,Zahl,Zeichen1,Zeichen2,Zeichen3
> SCHNEIDER CPC664,CPC6128  :
          KEY DEF Tastennr,Zahl,Zeichen1,Zeichen2,Zeichen3
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

KEY LIST

Befehlsart: Anweisung

KEY LIST

Beschreibung: Die mit KEY (siehe dort!) vorgenommene Funktionstastenbelegung wird auf dem Bildschirm angezeigt.
Dabei werden die Funktionen der einzelnen Funktionstasten der Reihe nach dargestellt.

Beispiel:

```
KEY LIST
color
auto
goto
list
run
color 15,4,4
cload"
cont
list.
 run
```

In dem Beispiel ist -color- der Inhalt der Funktionstaste 1 und -run- der der Funktionstaste 1Ø.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	KEY
COMMODORE 128	:	KEY
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	KEY LIST
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

KEY ON/OFF

Befehlsart: Anweisung

KEY ON
KEY OFF

Beschreibung: In der letzten Zeile des Bildschirmes eines MSX-Computers befindet sich eine Liste mit den Funktionstastenbelegungen (siehe KEY, siehe KEY LIST). Mit KEY ON kann man diese ein-, mit KEY OFF ausschalten.

Je nachdem, ob die SHIFT-Taste gedrückt ist, werden dabei die ersten fünf oder die zweiten fünf Funktionstastenbelegungen wiedergegeben.

Die Anzeige wird nicht nur mit KEY ON eingeschaltet, sondern auch beim Einschalten des Computers oder nach einem RESET (siehe dort!). Man kann in diesem Fall über eine Zeile weniger auf dem Bildschirm verfügen. Schalten wir die Anzeige mit KEY OFF wieder ab, steht uns diese Zeile wieder zur Verfügung.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	KEY ON / KEY OFF
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

KEY (X) ON/OFF/STOP

Befehlsart: Anweisung

KEY (**Zahl**) ON
KEY (**Zahl**) OFF
KEY (**Zahl**) STOP

Beschreibung: Anhand dieses Befehls können wir mit Hilfe der Funktionstaste Unterbrechungen herbeiführen und mit dem ON KEY GOSUB-Befehl (siehe ON KEY GOSUB) in ein Unterprogramm verzweigen.

Die **Zahl** gibt dabei an, für welche Funktionstaste eine Unterbrechung möglich ist. Sie kann Werte zwischen 1 und 1Ø annehmen.

KEY (**Zahl**) ON	Beim Drücken der Funktionstaste **Zahl** wird in das Unterprogramm verzweigt.
KEY (**Zahl**) OFF	Beim Drücken der Funktionstaste **Zahl** wird der normale Programmablauf nicht unterbrochen.
KEY (**Zahl**) STOP	Eine erneute Unterbrechung in der Unterbrechungsroutine wird verhindert, der Funktionstastendruck gespeichert, und nach der Ausführung eines RETURN-Befehles (siehe RETURN) wird die Unterbrechungsroutine erneut aufgerufen.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	KEY (Zahl) ON KEY (Zahl) OFF KEY (Zahl) STOP
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

LBOUND

Befehlsart: Funktion

LBOUND(**Variablenname**)
LBOUND(**Variablenname,Zahl**)

Beschreibung: Die Funktion gibt die untere Grenze eines durch DIM, NUMERIC oder STRING (siehe jeweils dort) definierten Feldes an.

Der **Variablenname** bezeichnet die Variable, die abgefragt werden soll.

Mit

LBOUND(**Variablenname**)

wird der kleinstmögliche Index bei einem eindimensionalen Feld abgefragt.

Mit

LBOUND(**Variablenname,Zahl**)

wird der kleinstmögliche Index bei einem Feld abgefragt, dessen Dimension die **Zahl** angibt. Ein dreidimensionales Feld hat drei Dimensionen; die **Zahl** kann somit die Werte 1, 2 oder 3 annehmen.

Beispiel:

```
PRINT LBOUND(A$)
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	LBOUND(Variable) LBOUND(Variable,Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

LCASE$

Befehlsart: Funktion

LCASE$(**String**)

Beschreibung: Mit dieser Funktion werden alle Buchstaben innerhalb eines **String**s in Kleinbuchstaben umgewandelt.
Die Funktion entspricht dem Befehl LOWER$ (siehe dort).

Beispiel:

```
PRINT LCASE("Schneider BUCH")
schneider buch
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	LCASE$(Textausdruck)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	LOWER$(Textausdruck)
SCHNEIDER CPC664,CPC6128	:	LOWER$(Textausdruck)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

LEFT$

Befehlsart: Funktion

LEFT$(**String,Zahl**)

Beschreibung: Mit dieser Funktion zur Wandlung von Zeichenketten erhält man einen Teilstring, der die durch die **Zahl** festgelegte Anzahl der linken Zeichen des **String**s enthält.

Beispiel:

```
10 A$="SCHNEIDER COMPUTER BUECHER"
20 PRINT MID$(A$,1,9)
RUN
SCHNEIDER
READY.
```

Anmerkungen zur Kompatibilität: Bei einigen BASIC-Dialekten fehlen die Funktionen LEFT$ und RIGHT$ (siehe dort). Sie sind dann zu ersetzen durch den Befehl MID$ (siehe dort).

Beispiel:

```
10 A$="SCHNEIDER COMPUTER BUECHER"
20 PRINT LEFT$(A$,9)
RUN
SCHNEIDER
READY.
```

Auch wenn es keine MID$-Funktion gibt, gibt es auch noch andere Möglichkeiten, um zur Wandlung von Zeichenketten LEFT$ und RIGHT$ zu simulieren.

So erfüllt bei den ATARI-Computern die Funktion

A$ = X$(1,**Zahl**)

und bei den SINCLAIR-Computern die Funktion

A$ = X$(1 TO **Zahl**)

die Funktion LEFT$(X$,**Zahl**).

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
› COMMODORE VC20,64	:	LEFT$(String,Zahl)
› COMMODORE 16,116,PLUS4	:	LEFT$(String,Zahl)
› COMMODORE 128	:	LEFT$(String,Zahl)
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	LEFT$(String,Zahl)
› SCHNEIDER CPC464	:	LEFT$(String,Zahl)
› SCHNEIDER CPC664,CPC6128	:	LEFT$(String,Zahl)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

LEN

Befehlsart: Funktion

LEN(**String**)
LEN("**Textausdruck**")

Beschreibung: Die Funktion LEN gibt die Zahl aller Zeichen wieder, die in einem **Textausdruck** bzw. einem definierten **String** enthalten sind. Auch die Leerstellen werden dabei mitgezählt.

Der Zahlenwert, der durch die Funktion LEN erzielt wird, kann zwischen Ø und – im Höchstfall – 255 liegen.

Beispiele:

```
10 REM*KREUZWORT-HILFE*
20 INPUT "DAS WORT";W$
30 PRINT "DAS WORT";W$
40 PRINT "HAT ";LEN (W$); "BUCHSTABEN."
RUN
DAS WORT? BASIC
DAS WORT BASIC
HAT  5 BUCHSTABEN.

READY.


PRINT LEN ("SCHNEIDER VERLAG")
 16

READY.
A$="SCHNEIDER VERLAG"

READY.
PRINT LEN (A$)
 16

READY.


10 INPUT A$
20 PRINT"□□"
30 FOR T=1 TO LEN(A$)+3
40 PRINT " ";
50 FOR K=1TO 250:NEXT
60 NEXT T:FOR L=1TO LEN (A$)+3:PRINT"■":NEXT L
70 PRINT "KANNST DU DAS NICHT ANDERS AUSDRUECKEN?":GOTO 10
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	LEN(String)
› COMMODORE VC20,64	:	LEN(String)
› COMMODORE 16,116,PLUS4	:	LEN(String)
› COMMODORE 128	:	LEN(String)
› ENTERPRISE 64,128	:	LEN(String)
› MSX-STANDARD	:	LEN(String)
› SCHNEIDER CPC464	:	LEN(String)
› SCHNEIDER CPC664,CPC6128	:	LEN(String)
› SINCLAIR ZX81	:	LEN String
› SINCLAIR SPECTRUM	:	LEN String

LET

Befehlsart: Anweisung

LET **Variablenname=Zahl**
LET **String = "Textausdruck"**

Beschreibung: Einer Zahlenvariablen wird ein bestimmter Wert zugewiesen, oder einer String-Variablen wird ein bestimmter **Textausdruck** zugeordnet.

Beispiele:

```
10 LET A=12
20 LET A$="MONATE"
30 LET B$="JAHR"
40 PRINT "DAS ";B$;" HAT ";A;" ";A$
```

```
LET A=10

PRINT A
 10
READY.
```

```
LET A$="SPEICHERKAPAZITAET"
PRINT A$
SPEICHERKAPAZITAET
READY.
```

Anmerkungen: Der Einsatz der Anweisung LET ist in den meisten BASIC-Dialekten nicht zwingend. Statt

```
LET A=3
```

genügt normalerweise

```
A=3
```

als Eingabe.

Die ausgeschriebene Anweisung LET ist ein Überbleibsel aus der Frühzeit der BASIC-Entwicklung. Sie wird bei den meisten Heimcomputern aus Gründen der Kompatibilität zugelassen. Jedoch ist die Anweisung selbst eine der wichtigsten Elemente beim Programmieren in BASIC, und zwar unabhängig davon, ob sie mit oder ohne LET verwendet wird.

Anmerkung zur Kompatibilität: Normalerweise wird für eine Variable, die nicht definiert worden ist, automatisch der Wert Ø angenommen. In einigen wenigen BASIC-Dialekten erscheint aber beim Einsatz einer vorher nicht definierten Variablen eine Fehlermeldung.

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :
› COMMODORE VC20,64         :
› COMMODORE 16,116,PLUS4    :
› COMMODORE 128             :
› ENTERPRISE 64,128         :  Bei allen diesen Computertypen erlaubt
› MSX-STANDARD              :
› SCHNEIDER CPC464          :
› SCHNEIDER CPC664,CPC6128  :
› SINCLAIR ZX81             :
› SINCLAIR SPECTRUM         :
```

LINE

Befehlsart: Anweisung

LINE STEP (**Zahl 1**,**Zahl 2**)- STEP(**Zahl 3**,**Zahl 4**),**Zahl 5**,**Zeichen**

Beschreibung: Mit diesem Befehl lassen sich Linien, bzw. Rechtecke auf dem Bildschirm darstellen.

Die STEP-Option wird verwendet, wenn man die Koordinaten der Geraden in Beziehung zur letzten Stellung des Grafik-Cursors angibt. Sie kann auch entfallen.

Zahl 1 und **Zahl 3** geben die X-Richtungen der Endpunkte der Geraden an. **Zahl 2** und **Zahl 4** geben die Y-Richtungen der Endpunkte der Geraden an. **Zahl 5** bezeichnet die Farbnummer.

Mit dem **Zeichen** legt man fest, welche Funktion der Befehl hat. Läßt man es weg, wird eine Gerade gezeichnet. Ist es ein B, so wird ein Rechteck gezeichnet, wobei die Eckpunkte durch

Zahl 1,**Zahl 2**
Zahl 1,**Zahl 4**
Zahl 3,**Zahl 2**
Zahl 3,**Zahl 4**

gekennzeichnet werden.

Ist das **Zeichen** ein BF, wird das Rechteck mit der angegebenen Farbe ausgemalt.

Das Zeichen verwendet man ohne Anführungsstriche.

Beispiel:

```
10 SCREEN 2
20 FOR N=300 TO 0 STEP -20
30 LINE (0,0) - (N,N),((N/10)AND 15),BF
40 NEXT
50 GOTO 50
```

Kompatibilitätsliste:

```
ATARI 600,800,130XE       :
COMMODORE VC20,64         :
COMMODORE 16,116,PLUS4    :
    DRAW Zahl5,Zahl1,Zahl2,Zahl3,Zahl4
COMMODORE 128             :
    DRAW Zahl5,Zahl1,Zahl2,Zahl3,Zahl4
ENTERPRISE 64,128         :
> MSX-STANDARD            :
    LINE STEP(Zahl1,Zahl2)-STEP(Zahl3,Zahl4),Zahl5,Zeichen
SCHNEIDER CPC464          :
SCHNEIDER CPC664,CPC6128  :
SINCLAIR ZX81             :
SINCLAIR SPECTRUM         :
```

LINE INPUT

Befehlsart: Anweisung

LINE INPUT **Kanalnummer,;"Textausdruck";Stringvariable**

Beschreibung: Dieser Befehl ist mit der INPUT-Anweisung (siehe INPUT) eng verwandt. Es bestehen jedoch einige entscheidende Unterschiede:

- Man kann nur eine **Stringvariable** pro Befehl verwenden.
- Beim LINE INPUT-Befehl wird im Gegensatz zu INPUT kein Fragezeichen auf dem Bildschirm gezeigt.
- Beim LINE INPUT-Befehl dürfen bei der Texteingabe - anders als bei INPUT - auch Anführungszeichen, Kommata, Doppelpunkte usw. verwendet werden.

Zur Befehlssyntax: **Kanalnummer** muß nur angegeben werden, wenn man statt Bildschirm und Tastatur andere Ein-/Ausgabegeräte verwenden will. Der Strichpunkt (;) vor dem "**Textausdruck**" muß nur eingegeben werden, wenn man nach der Eingabe keinen Zeilenvorschub (= neue Zeile) wünscht.

"**Textausdruck**" muß nur dann eingegeben werden, wenn man Hinweise für die Eingabe einfügen will, wie z. B. "Geben Sie bitte ein". Der vom Benutzer eingegebene Text wird der **Stringvariablen** übergeben. Man kann diese später mit den String-Befehlen wie LEFT$, RIGHT$, LOWER$ usw. (siehe jeweils dort) bearbeiten oder mit dem PRINT-Befehl (siehe dort) ausgeben lassen.

Beispiel:

```
10 LINE INPUT "Geben Sie bitte einen Text ein:";a$
20 PRINT:PRINT "Der eingegebene Text lautet:"
30 PRINT a$
```

Anmerkung: Weil der Benutzer bei Verwendung dieses Befehls auch Satzzeichen eingeben darf, eignet sich der Befehl besonders für ein Textverarbeitungsprogramm.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	LINE INPUT #Zahl, PROMPT Text:Stringvariable
› MSX-STANDARD	:	LINE INPUT Text;Stringvariable
› SCHNEIDER CPC464	:	LINE INPUT #Zahl,Text;Stringvariable
› SCHNEIDER CPC664,CPC6128	:	LINE INPUT #Zahl,Text;Stringvariable
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

LIST

Befehlsart: Kommando

LIST
LIST **Zeilennummer 1-Zeilennummer 2** oder
LIST **Zeilennummer 1,Zeilennummer 2**
LIST **Zeilennummer**
LIST **-Zeilennummer**
LIST **Zeilennummer-**

Beschreibung: Mit dem Kommando LIST bringen wir das BASIC-Programm auf den Bildschirm, sofern darin kein LIST-Schutz enthalten ist. Durch LIST ist eine vollständige Ausgabe des Programms möglich, bei Angabe von **Zeilennummern** aber stets auch eine teilweise Ausgabe.

Dabei bedeutet zum Beispiel

LIST 1ØØ-15Ø das Auflisten der Programmzeilen 1ØØ bis einschließlich 15Ø;

LIST 21Ø das Auflisten der Programmzeile 21Ø;

LIST -2ØØ das Auflisten des Programmes von Programmanfang bis Programmzeile 2ØØ;

LIST 122Ø- das Auflisten des Programmes von Programmzeile 122Ø bis Programmende;

LIST das Auflisten des gesamten BASIC-Programmes.

Beispiel:

```
190 IF B=6 OR B=7 OR B=8 THEN PRINT "STI
MMT.DER SOMMER IST AM SCHOENSTEN.":END
200 IF B=9 OR B=10 OR B=11 THEN PRINT"ST
IMMT.DER HERBST IST AM SCHOENSTEN.":END
210 INPUT "DER WIEVIELTE DEZEMBER";C
220 IF C<10 THEN PRINT "NOCH EIN BISSCHE
N GEDULD.":END
230 IF C<20 THEN PRINT "JETZT WIRD ES SC
HON LANGSAM SPANNEND.":END
240 IF C<25 THEN PRINT "EIN SCHOENES FES
T WUENSCHT AUCH DER C64!":END
250 PRINT "WAR DAS WEIHNACHTSFEST SCHOEN
?"

LIST 115-135

120 IF A=11 THEN PRINT "ACH, WIR HABEN N
OVEMBER!":GOTO 150
130 IF A=12 THEN PRINT "ACH, WEIHNACHTSZ
EIT!":GOTO 151
```

Anmerkung: Bei der Eingabe nicht vorhandener Programmzeilen wird die nächsthöhere bzw. nächstniedrige **Zeilennummer** angenommen.

Beispiel: Bei einem Programm, das in Zehnerschritten numeriert ist, führt das Kommando

```
LIST 15-45
```

zum Auflisten der Programmzeilen 2Ø - 4Ø.

Nach der Ausführung des LIST-Kommandos befindet sich der Computer im Direkt-Modus. In die Ausführung des LIST-Kommandos läßt sich durch RUN/STOP, BREAK, ESC usw. eingreifen.

Besonderheit: Beim C 64 wird durch Drücken der CTRL-Taste die Ausführung verlangsamt; das heißt, das Listing rollt im Bildschirm langsamer nach oben.

Das Kommando LIST kann man in ein Programm "einbauen", aber das Programm wird danach abgebrochen.

Beispiel:

```
10 PRINT "GIB ZWEI ZAHLEN EIN.DIE ERSTE GROESSER,"
20 PRINT "DIE ZWEITE KLEINER. ABER NICHT"
30 PRINT "KLEINER ALS FUENF."
40 INPUT A
50 INPUT B
60 IF A>B AND B>5 THEN C=A-B:GOTO 80
70 PRINT "FALSCHE EINGABE. NOCHMAL:":GOTO 40
80 PRINT "WIEVIEL IST ";A;" - ";B
90 INPUT D
100 IF D=C THEN PRINT "RICHTIG!":GOTO 120
110 PRINT "LEIDER FALSCH. NOCHMAL:":GOTO90
120 PRINT "       WILLST DU WISSEN, WIE"
130 PRINT "      DAS PROGRAMM AUFGEBAUT"
140 INPUT "IST? DANN DRUECKE >J<";R$
150 IF R$="J" THEN LIST
```

Anmerkung zur Kompatibilität: In einigen BASIC-Dialekten wird zum Auflisten eines Programmteiles statt des - (das Zeichen für "bis") ein Komma verwendet. Dann führt LIST 12Ø,18Ø zum Auflisten der Programmzeilen von 12Ø bis einschließlich 18Ø.

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  LIST Zeilenummer1-Zeilennummer2
> COMMODORE VC20,64         :  LIST Zeilenummer1-Zeilennummer2
> COMMODORE 16,116,PLUS4    :  LIST Zeilenummer1-Zeilennummer2
> COMMODORE 128             :  LIST Zeilenummer1-Zeilennummer2
> ENTERPRISE 64,128         :  LIST Zeilenummer1 TO Zeilennummer2
> MSX-STANDARD              :  LIST Zeilenummer1-Zeilennummer2
> SCHNEIDER CPC464          :  LIST Zeilenummer1-Zeilennummer2
> SCHNEIDER CPC664,CPC6128  :  LIST Zeilenummer1-Zeilennummer2
> SINCLAIR ZX81             :  LIST Zeilenummer1-Zeilennummer2
> SINCLAIR SPECTRUM         :  LIST Zeilenummer1-Zeilennummer2
```

LLIST

Befehlsart: Kommando

LLIST
LLIST Zeilennummer-Zeilennummer

Beschreibung: Das Kommando wird entsprechend dem Kommando LIST (siehe dort) ausgeführt. Doch wird das Programmlisting nicht auf den Bildschirm gebracht, sondern vom angeschlossenen Drucker aufgenommen und ausgedruckt.

Beispiel:

```
LLIST 120
LLIST 340-450
LLIST -80
LLIST 3450-
LLIST
```

Anmerkung: Wie LIST kann man LLIST auch innerhalb eines Programms verwenden.

Beispiel:

```
100 PRINT "WOLLEN SIE EINEN AUSDRUCK"
110 INPUT "DES PROGRAMMES"; A$
120 IF A$="JA"THEN LLIST
```

Anmerkung zur Kompatibilität: Eine ähnliche Funktion haben in anderen BASIC-Dialekten Kommandos wie LIST/P oder LIST#**Zahl** (Schneider CPC), wobei die **Zahl** der Adresse des Druckers entspricht (siehe Kompatibilitätsliste).

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
        OPEN1,4,0:CMD1:LIST Zeilenummer1-Zeilennummer2
  COMMODORE 16,116,PLUS4    :
        OPEN1,4,0:CMD1:LIST Zeilenummer1-Zeilennummer2
  COMMODORE 128             :
        OPEN1,4,0:CMD1:LIST Zeilenummer1-Zeilennummer2
> ENTERPRISE 64,128         :   LLIST Zeilenummer1 TO Zeilennummer2
> MSX-STANDARD              :   LLIST Zeilenummer1-Zeilennummer2
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
> SINCLAIR ZX81             :   LLIST Zeilenummer1-Zeilennummer2
> SINCLAIR SPECTRUM         :   LLIST Zeilenummer1-Zeilennummer2
```

LN

Befehlsart: Funktion

LN **Zahl**

Beschreibung: Der Logarithmus einer **Zahl** wird zurückgegeben. Diese Funktion entspricht der Funktion LOG (siehe dort).

Beispiel:

```
PRINT LN 1
 Ø
READY
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :  LOG(Zahl)
  COMMODORE VC20,64         :  LOG(Zahl)
  COMMODORE 16,116,PLUS4    :  LOG(Zahl)
  COMMODORE 128             :  LOG(Zahl)
  ENTERPRISE 64,128         :  LOG(Zahl)
  MSX-STANDARD              :  LOG(Zahl)
  SCHNEIDER CPC464          :  LOG(Zahl)
  SCHNEIDER CPC664,CPC6128  :  LOG(Zahl)
> SINCLAIR ZX81             :  LN Zahl
> SINCLAIR SPECTRUM         :  LN Zahl
```

LOAD

Befehlsart: Kommando

LOAD
LOAD "Programmname"
LOAD "Programmname",(Zahl)

Beschreibung: Mit LOAD kann man ein Programm von einem Peripherie-Gerät in den Computer laden. Das unter dem "Programmnamen" abgespeicherte Programm wird gesucht und geladen. Bei der Eingabe des "Programmnamens" sind auch die Leerstellen zu beachten. Fehlt der "Programmname", wird im allgemeinen, wenn es sich um einen Ladevorgang von Kassette handelt, das erste gefundene Programm geladen.

Falls die Programme nicht entsprechend konstruiert sind, wird ein vorher im Computer geladenes Programm durch das mit LOAD neu geladene Programm gelöscht und alle Datenfiles werden geschlossen.

Das Format des LOAD-Kommandos ist von Computertyp zu Computertyp unterschiedlich. Die (Zahl) kann dabei der Adresse des Peripherie-Gerätes entsprechen. Die Commodore-Computer nehmen bei fehlender (Zahl) Geräteadresse 1, also die des Kassettenrecorders an.

Manchmal ist es auch möglich, Daten-Felder zu speichern und zu laden, etwa beim Sinclair-BASIC. Das entsprechende Kommando hat dann das Format

LOAD "Dateiname" DATA A(Zahl).

Beim Schneider-Computer CPC464 und CPC664 läßt sich eine binäre Datei in den Speicher des Computers laden, sofern das Kommando folgendes Format hat:

LOAD "Dateiname"(Adressenausdruck)

LOAD kann man auch innerhalb eines Programmes verwenden.

Beispiel:

```
10 INPUT "Dein Name";na$
20 PRINT "Wollen wir ein Spiel machen, ";na$;"?"
30 PRINT "Gebe >ja< oder >nein< ein."
40 INPUT an$
50 IF an$="ja" THEN LOAD "schlangenspiel"
```

Kompatibilitätsliste:

Laden einer Kassette

> ATARI 600,800,130XE	:	LOAD "C:Programmname" CLOAD "Programmname"
> COMMODORE VC20,64	:	LOAD "Programmname"
> COMMODORE 16,116,PLUS4	:	LOAD "Programmname"
> COMMODORE 128	:	LOAD "Programmname"
> ENTERPRISE 64,128	:	LOAD "Programmname"
> MSX-STANDARD	:	LOAD "CAS:Programmname" CLOAD "Programmname"
> SCHNEIDER CPC464	:	LOAD "Programmname"
SCHNEIDER CPC664,CPC6128	:	
> SINCLAIR ZX81	:	LOAD "Programmname"
> SINCLAIR SPECTRUM	:	LOAD "Programmname"

Laden einer Diskette

> ATARI 600,800,130XE	:	LOAD "D:Programmname"
> COMMODORE VC20,64	:	LOAD "Programmname",Zahl
> COMMODORE 16,116,PLUS4	:	LOAD "Programmname",Zahl DLOAD "Programmname",D0,UZahl
> COMMODORE 128	:	LOAD "Programmname",Zahl DLOAD "Programmname",D0,UZahl
> ENTERPRISE 64,128	:	LOAD "Programmname"
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
> SCHNEIDER CPC664,CPC6128	:	LOAD "Programmname"
SINCLAIR ZX81	:	
> SINCLAIR SPECTRUM	:	LOAD*"M";Laufwerksnummer;"Programmname"

LOCATE

Befehlsart: Anweisung

LOCATE **Zahl 1,Zahl 2,Zahl 3**

Beschreibung: Mit diesem Befehl läßt sich der Textcursor an eine bestimmte Stelle des Bildschirms setzen, um dort einen Textausdruck erscheinen zu lassen.

Zahl 1 bezeichnet dabei das Ein-/Ausgabegerät. Falls mit dem Bildschirm gearbeitet wird, kann sie fehlen, so daß der Befehl das Format hat

LOCATE **Zahl 2,Zahl 3**

Zahl 2 bezeichnet die Bewegung auf der x-Achse (Spalte).
Zahl 3 bezeichnet die Bewegung auf der y-Achse (Zeile).

Dabei ist zu beachten, daß der Ursprung ("Nullpunkt") des Textcursors stets die linke obere Ecke des Bildschirmes bzw. eines "Fensters" (siehe WINDOW) ist und bleibt. Zudem ist darauf zu achten, in welchem Bildschirm-Modus (siehe MODE) man sich befindet. Die x-Koordinate kann dann maximal entweder 2Ø, 4Ø oder 8Ø als Wert annehmen.

Wird dieser maximale Wert überschritten - gleichgültig ob um 1 oder mehr -, so erscheint der Textausdruck am Beginn der nächsten Zeile.

Wird hingegen der Wert für die y-Koordinate überschritten (23 - 25), wird der Text in die letzte Zeile geschrieben, gleichgültig um welchen Wert er überschritten wurde. Die Spaltenverschiebung wird jedoch beachtet.

Nur bei sehr hohem Überschreiten des Zeilen-Wertes wird der Textausdruck unter Beachtung der Spaltenverschiebung in die erste Zeile (des Bildschirms) gebracht.

Beispiel:

```
10 CLS
20 LOCATE 10,20
30 PRINT "a"
40 LOCATE 5,10
50 PRINT "b"
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	POSITION Zahl2,Zahl3
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	LOCATE Zahl2,Zahl3
› COMMODORE 128	:	LOCATE Zahl2,Zahl3
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	LOCATE Zahl2,Zahl3,Zahl4
› SCHNEIDER CPC464	:	LOCATE #Zahl1,Zahl2,Zahl3
› SCHNEIDER CPC664,CPC6128	:	LOCATE #Zahl1,Zahl2,Zahl3
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

LOG

Befehlsart: Funktion

LOG(**Zahl**)

Beschreibung: Mit dieser mathematischen Funktion wird der natürliche Logarithmus (d. h. zur Basis e) wiedergegeben.

Beispiel:

```
? LOG(13)
 2.56494936
READY
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	LOG(Zahl)
› COMMODORE VC20,64	:	LOG(Zahl)
› COMMODORE 16,116,PLUS4	:	LOG(Zahl)
› COMMODORE 128	:	LOG(Zahl)
› ENTERPRISE 64,128	:	LOG(Zahl)
› MSX-STANDARD	:	LOG(Zahl)
› SCHNEIDER CPC464	:	LOG(Zahl)
› SCHNEIDER CPC664,CPC6128	:	LOG(Zahl)
SINCLAIR ZX81	:	LN Zahl
SINCLAIR SPECTRUM	:	LN Zahl

LOG2

Befehlsart: Funktion

LOG2(**Zahl**)

Beschreibung: Mit dieser Funktion erhält man den Logarithmus einer **Zahl** zur Basis 2. Die **Zahl** darf dabei nicht negativ sein.

Beispiel:

```
PRINT LOG2(2)
 1
READY
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	LOG2(Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

LOG1Ø

Befehlsart: Funktion

LOG1Ø(**Zahl**)

Beschreibung: Mit dieser mathematischen Funktion wird der dekadische Logarithmus (d. h. zur Basis 1Ø) wiedergegeben.

Beispiel:

```
? LOG1Ø (13)
 1.11394335
READY
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	CLOG(Zahl)
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	LOG10(Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

LOOK

Befehlsart: Anweisung

LOOK **Kanalnummer** AT **Zahl 1,Zahl 2:Variablenname**
LOOK AT **Zahl 1,Zahl 2:Variablenname**
LOOK **Variablenname**

Beschreibung: Der Zahlenvariablen mit dem **Variablennamen** wird die Farbe der aktuellen Stellung des Grafik-Cursors zugewiesen.

Gibt man im ersten Format des Befehls **Zahl 1** und **Zahl 2** an, wird die Farbe der durch diese Zahlen festgelegten Cursor-Stellung abgefragt.

Mit Hilfe der **Kanalnummer** läßt sich die Farbe der Cursor-Stellung einer augenblicklich nicht angezeigten Grafikseite abfragen.

Beispiel:

```
10 GRAPHICS
20 FOR I=1 TO 200
30 LOOK AT I,100:J
40 PRINT J
50 NEXT I
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :
        LOOK #Kanalnummer AT Zahl1,Zahl1:Variable
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

LOOP

(siehe unter DO LOOP)

LOWER$

Befehlsart: Funktion

LOWER$(**String**)
LOWER$("**Textausdruck**")

Beschreibung: Lower$ gibt den angegebenen **Textausdruck** ausschließlich in Kleinbuchstaben wieder, falls dieser in Großbuchstaben oder in gemischter Schreibweise eingegeben wurde. Mit dieser Funktion, bzw. ihrer Umkehrung UPPER$ (siehe dort), lassen sich Eingaben auswerten und vergleichen, die in unterschiedlichen Schreibweisen eingegeben wurden.

Beispiel:

```
10 INPUT a$
20 PRINT LOWER$(a$)
Schneider Computer Buecher
schneider computer buecher
```

Anmerkung: Die Funktion betrifft ausschließlich Buchstaben. Satzzeichen und Sonderzeichen sind davon nicht be-

troffen. Bei den Enterprise-Rechnern heißt dieser Befehl LCASE$ (siehe dort).

Kompatibilitätsliste:

	ATARI 600,800,130XE	:
	COMMODORE VC20,64	:
	COMMODORE 16,116,PLUS4	:
	COMMODORE 128	:
	ENTERPRISE 64,128	: LCASE$(Textausdruck)
	MSX-STANDARD	:
›	SCHNEIDER CPC464	: LOWER$(Textausdruck)
›	SCHNEIDER CPC664,CPC6128	: LOWER$(Textausdruck)
	SINCLAIR ZX81	:
	SINCLAIR SPECTRUM	:

LPOS

Befehlsart: Funktion

LPOS(**Zahl**)

Beschreibung: Für den Schreibkopf des angeschlossenen Druckers wird die Spaltenposition bestimmt. Die **Zahl** ist dabei ein Formatparameter, den der Rechner ermittelt.

In unserem Beispiel gibt der Rechner mit der PRINT-Anweisung (siehe dort) für LPOS eine 1Ø zurück.

Beispiel:

```
10 LPRINT "1234567890";
20 PRINT LPOS(X)
```

Kompatibilitätsliste:

	ATARI 600,800,130XE	:
	COMMODORE VC20,64	:
	COMMODORE 16,116,PLUS4	:
	COMMODORE 128	:
	ENTERPRISE 64,128	:
›	MSX-STANDARD	: LPOS(Zahl)
	SCHNEIDER CPC464	:
	SCHNEIDER CPC664,CPC6128	:
	SINCLAIR ZX81	:
	SINCLAIR SPECTRUM	:

LPRINT

Befehlsart: Kommando

LPRINT "**Textausdruck**"
LPRINT **Stringvariable**
LPRINT **Rechenausdruck**

Beschreibung: Das LPRINT-Kommando entspricht dem Kommando PRINT (siehe dort) mit dem Unterschied, daß der **Textausdruck** oder das Ergebnis des **Rechenausdrucks** statt auf dem Bildschirm direkt über den Drucker ausgegeben wird.

Beispiel:

```
10 INPUT N
20 LPRINT "NUMMER";N
30 INPUT "NAME";NA$
40 LPRINT "NAME";NA$
```

Kompatibilitätsliste:

›	ATARI 600,800,130XE	LPRINT	Ausdruck
	COMMODORE VC20,64		
	COMMODORE 16,116,PLUS4		
	COMMODORE 128		
›	ENTERPRISE 64,128	LPRINT	Ausdruck
›	MSX-STANDARD	LPRINT	Ausdruck
	SCHNEIDER CPC464		
	SCHNEIDER CPC664,CPC6128		
	SINCLAIR ZX81		
›	SINCLAIR SPECTRUM	LPRINT	Ausdruck

LIRIM$

Befehlsart: Funktion

LTRIM$(**String**)

Beschreibung: Mit dieser Funktion werden alle Leerzeichen am Anfang eines **Strings** entfernt.

Beispiel:
```
PRINT LTRIM$("    PETER")
PETER
READY
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	LTRIM$(String)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

MAX

Befehlsart: Funktion

MAX(**Zahlenliste**)

Beschreibung: Es wird die größte Zahl einer **Zahlenliste** ermittelt. Die Zahlen in der **Zahlenliste** werden durch Kommata getrennt und stehen in Klammern. Statt einer Liste mit Zahlen kann man auch die Liste mit Variablen verwenden. Die Variablen müssen allerdings numerische Variablen, also Zahlenvariablen, sein.

Beispiel:
```
? MAX (13,2,4,8,Ø,-12,-16)
 13
READY
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	MAX(Zahlenliste)
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	MAX(Zahlenliste)
› SCHNEIDER CPC664,CPC6128	:	MAX(Zahlenliste)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

MAXLEN

Befehlsart: Funktion

MAXLEN(Variablenname)

Beschreibung: Die Funktion gibt dem Benutzer die maximale Länge einer von ihm definierten Stringvariablen mit dem Variablennamen an.

Beispiel:

```
PRINT MAXLEN(A$)
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
> ENTERPRISE 64,128	:	MAXLEN(Variable)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

MEMORY

Befehlsart: Kommando

MEMORY **Adresse**

Beschreibung: Mit dem MEMORY-Befehl läßt sich die Obergrenze im RAM-Speicher festlegen, der für BASIC zur Verfügung stehen soll. Damit läßt sich auf sehr einfache Weise Speicherplatz, zum Beispiel für Maschinenspracheprogramme, schaffen.

Bei Überschreiten der Obergrenze erscheint die Fehlermeldung

MEMORY FULL.

Welche Obergrenze festgelegt wurde, kann man jederzeit mit HIMEM (siehe dort) abfragen.

Beispiel:

```
MEMORY 3Ø
PRINT HIMEM
3Ø
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
› SCHNEIDER CPC464          :  MEMORY Adresse
› SCHNEIDER CPC664,CPC6128  :  MEMORY Adresse
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

MERGE

Befehlsart: Kommando

MERGE **Programmname**

Beschreibung: Das Programm mit dem **Programmnamen** wird in den Rechnerspeicher geladen, ohne daß das im Speicher befindliche Programm gelöscht wird. Es werden nur diejenigen Programmzeilen gelöscht, deren Zeilennummern mit den Zeilennummern des neuen Programmes übereinstimmen.

Beispiel:

MERGE "druckertreiber"

Anmerkung: Durch MERGE wird die Wirkung ausgeführter Befehle wie ON ERROR GOTO, DEF FN, DEFINT, DEFREAL, DEFSTR (siehe jeweils dort) aufgehoben. Zahlenvariablen erhalten den Wert Ø, Stringvariablen werden zu Leerstrings (""). Offene Dateien werden gelöscht.

Mit Hilfe dieses Befehls kann der erfahrene BASIC-Programmierer sich einzelne Programmroutinen, die er des öfteren verwendet, zu einem Programm zusammen-"mergen" (to merge = englisch für "verschmelzen").

Kompatibilitätsliste:

```
  ATARI 600,800,130XE          :  ENTER "Programmname"
  COMMODORE VC20,64            :
  COMMODORE 16,116,PLUS4       :
  COMMODORE 128                :
> ENTERPRISE 64,128            :  MERGE "Programmname"
> MSX-STANDARD                 :  MERGE "CAS:Programmname"
> SCHNEIDER CPC464             :  MERGE "Programmname"
> SCHNEIDER CPC664,CPC6128     :  MERGE "Programmname"
  SINCLAIR ZX81                :
> SINCLAIR SPECTRUM            :  MERGE "Programmname"
                                  MERGE*"M";Laufwerksnummer;"Programmname"
```

MID$

Befehlsart: Anweisung

MID$(String 1,Zahl 1,Zahl 2)=String 2

Beschreibung: Mit dieser Anweisung kann man einen Teilstring einer Zeichenkette austauschen.

Dabei wird durch **String 1,Zahl 1** und **Zahl 2** festgelegt, welcher Stringteil ausgetauscht werden soll.

String 1 ist der String, der verändert werden soll. Es kann sich dabei nur um eine Variable handeln.

Zahl 1 gibt die erste Zeichenstelle an, an der der neue String stehen soll.

Zahl 2 ist die Anzahl der Zeichen, die ersetzt werden sollen.

String 2 ist der neue Stringteil. Falls dieser länger als die durch **Zahl 2** bezeichnete Zeichenanzahl ist, werden nur die ersten benötigten Zeichen verwendet.

Beispiel:

```
10 A$="GELD"
20 MID$(A$,2,1)="O"
30 PRINT A$
```

Anmerkung zur Kompatibilität: Die Anweisung MID$ ist nicht mit der Funktion MID$ (siehe dort) zu verwechseln, die einen Teilstring des ursprünglichen Strings zurückgibt.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  MID$(String1,Zahl1,Zahl2)=String2
> COMMODORE 128             :  MID$(String1,Zahl1,Zahl2)=String2
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  MID$(String1,Zahl1,Zahl2)=String2
  SCHNEIDER CPC464          :
> SCHNEIDER CPC664,CPC6128  :  MID$(String1,Zahl1,Zahl2)=String2
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

MID$

Befehlsart: Funktion

MID$(**String,Zahl 1,Zahl 2**)

Beschreibung: Mit dieser Funktion zur Wandlung von Zeichenketten erhält man einen Teilstring. Er beginnt mit dem von **Zahl 1** bezeichneten Zeichen des **String**s und enthält so viele Zeichen, wie die **Zahl 2** angibt.

Beispiel:
```
10 A$="SCHNEIDER COMPUTER BUECHER"
20 PRINT MID$(A$,11,8)
RUN
COMPUTER

READY.
```

Anmerkung: Die MID$-Funktion läßt sich auch in Verbindung mit Schleifen einsetzen.

Beispiel:
```
10 A$="SCHNEIDER COMPUTER BUECHER"
20 FOR I=1 TO LEN(A$)
30 PRINT MID$(A$,I,1);
40 FOR T=1 TO 200:NEXT
50 NEXT I
```

Anmerkung zur Kompatibilität: In einigen BASIC-Dialekten wird diese Funktion geschrieben als SEG$.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
› COMMODORE VC20,64	:	MID$(String,Zahl1,Zahl2)
› COMMODORE 16,116,PLUS4	:	MID$(String,Zahl1,Zahl2)
› COMMODORE 128	:	MID$(String,Zahl1,Zahl2)
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	MID$(String,Zahl1,Zahl2)
› SCHNEIDER CPC464	:	MID$(String,Zahl1,Zahl2)
› SCHNEIDER CPC664,CPC6128	:	MID$(String,Zahl1,Zahl2)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

MIN

Befehlsart: Funktion

MIN(**Zahlenliste**)

Beschreibung: Es wird die kleinste Zahl einer **Zahlenliste** ermittelt. Die Zahlen in der **Zahlenliste** werden durch Kommata getrennt und stehen in Klammern. Statt einer Liste mit Zahlen kann man auch eine Liste mit Variablen verwen-

den. Die Variablen müssen allerdings numerische Variablen, also Zahlenvariablen, sein.

Beispiel:

```
? MIN (13,2,4,8,Ø,-12,-16)
-16
READY
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	MIN(Zahlenliste)
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	MIN(Zahlenliste)
› SCHNEIDER CPC664,CPC6128	:	MIN(Zahlenliste)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

MOD

Befehlsart: Operand

Zahl 1 MOD **Zahl 2**

Beschreibung: Mit dem Operanden MOD wird der Rest einer Division (Teilung) wiedergegeben, der nach dem ganzzahligen Ergebnis bleibt.

Nach dem Rest-Verfahren kommt bei der Teilung von 1Ø durch 3 das Ergebnis 3 Rest 1 heraus. Eben diesen Rest erhält man mit dem Operanden MOD.

Zahl 1 ist die Zahl, die geteilt wird (Zähler, Dividend).

Zahl 2 ist die Zahl, durch die geteilt wird (Nenner, Divisor).

Beispiel: PRINT 1Ø MOD 3
 1
 READY

Anmerkung zur Kompatibilität: Bei Rechnern, deren BASIC über diesen Operanden nicht verfügt, kann der Befehl wie folgt umschrieben werden:

Zahl 1 MOD **Zahl 2**

entspricht
für Zahlen mit gleichem Vorzeichen (z. B. beide Zahlen positiv):

Zahl 1-INT(**Zahl 1**/**Zahl 2**)*__Zahl 2__

und für Zahlen mit unterschiedlichem Vorzeichen (z. B. **Zahl 1** positiv, **Zahl 2** negativ):

Zahl 1+INT(-**Zahl 1**/**Zahl 2**)*__Zahl 2__

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	MOD(Zahl1,Zahl2)
› MSX-STANDARD	:	Zahl1 MOD Zahl2
› SCHNEIDER CPC464	:	Zahl1 MOD Zahl2
› SCHNEIDER CPC664,CPC6128	:	Zahl1 MOD Zahl2
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

MODE

Befehlsart: Kommando

MODE **Zahl**

Beschreibung: Die CPCs besitzen drei Bildschirmmodi, die sich durch unterschiedliche Auflösung - also auch unter-

schiedliche Schriftgrößen - unterscheiden. Der Bildschirmmodus wird durch den Befehl MODE, gefolgt von einer **Zahl**, eingestellt, deren Wert Ø, 1 oder 2 betragen kann. MODE Ø stellt den Betrieb mit der geringsten Auflösung dar, nämlich 2Ø Zeichen pro Zeile.

MODE 1 ist der normale Betriebsmodus, in dem der Rechner automatisch nach dem Einschalten arbeitet. Die Auflösung beträgt 4Ø Zeichen pro Zeile.

MODE 2 schließlich bietet die höchste Auflösung mit 8Ø Zeichen pro Zeile.

Beispiel:

```
10 MODE 0
20 PRINT "schneider"
30 PRINT "computer"
40 PRINT "buch"
50 GOTO 50
```

Anmerkungen: Zu beachten ist bei dem Wechsel von einem Bildschirmmodus in den anderen, daß mit zunehmender Auflösung die Anzahl der zur Verfügung stehenden Farben stark abnimmt.

Bei jedem Wechsel wird zudem der Cursor in die linke obere Ecke des Bildschirms (Startpunkt des Cursors) gesetzt. Dabei wird auch der Bildschirm gelöscht.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	GRAPHICS Zahl1
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	GRAPHIC Zahl1,Zahl2,Zahl3
COMMODORE 128	:	GRAPHIC Zahl1,Zahl2,Zahl3
ENTERPRISE 64,128	:	TEXT Zahl
MSX-STANDARD	:	SCREEN Zahl1
› SCHNEIDER CPC464	:	MODE Zahl1
› SCHNEIDER CPC664,CPC6128	:	MODE Zahl1
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

MONITOR

Befehlsart: Kommando

MONITOR

Beschreibung: Der im Rechner integrierte Maschinensprache-Monitor wird aufgerufen.

Anmerkung: Der Befehl ist nur von Bedeutung, wenn man in Maschinensprache programmiert. Für BASIC-Programmierung ist er unerheblich.

Der Vollständigkeit halber hier die Befehle des Monitors (ausführlichere Erklärungen sind im entsprechenden Handbuch der Computer Commodore C16, C116, C128 und des Plus4 zu finden):

A (assemble)	Assembliert eine Programmzeile
C (compare)	Vergleicht zwei Speicherbereiche
D (disassemble)	Disassembliert einen Speicherbereich
F (fill)	Füllt einen Speicherbereich mit einem Byte
G (go)	Startet ein Maschinenprogramm
H (hunt)	Sucht in einem Speicherbereich nach einer Byte-Folge
L (load)	Lädt ein Maschinenprogramm
M (memory)	Zeigt den Inhalt eines Speicherbereiches an
R (register)	Zeigt den Inhalt des Prozessorregisters an
S (save)	Speichert einen bestimmten Speicherbereich
T (transfer)	Verschiebt einen bestimmten Speicherbereich
V (verify)	Überprüft eine Abspeicherung
X (exit)	Kehrt in BASIC zurück

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  MONITOR
> COMMODORE 128             :  MONITOR
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

MOTOR

Befehlsart: Anweisung

MOTOR
MOTOR ON
MOTOR OFF

Beschreibung: Der Motor eines angeschlossenen Kassettenrecorders wird eingeschaltet (MOTOR ON) oder abgeschaltet (MOTOR OFF).
Der Befehl

MOTOR

bewirkt, daß vom gerade wirksamen Modus in den jeweils anderen umgeschaltet wird.

Beispiel:

```
11Ø  INPUT AN$
12Ø IF AN$="J" THEN MOTOR ON
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  MOTOR ON       /      MOTOR OFF
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

MOVE

Befehlsart: Anweisung

MOVE **Zahl 1,Zahl 2**

Beschreibung: Mit diesem Befehl läßt sich der Grafik-Cursor bewegen, ohne daß ein Punkt, eine Linie oder andere sichtbare Zeichen auf den Bildschirm gebracht werden. Dabei gibt **Zahl 1** die X-Koordinate, **Zahl 2** die Y-Koordinate für die Bewegung wieder.

Der Befehl

```
MOVE 2Ø,3Ø
```

bewegt den Grafik-Cursor von Punkt a (alte Position) zu Punkt n (neue Position).

Wenn die alte Position des Cursors a ist, wird dieser mit MOVE 4Ø,4Ø auf die Position n gebracht.

Will man den Befehl MOVE einsetzen, muß vorher der Grafik-Bildschirm aufgerufen werden. Das geschieht mit MODE2 (siehe unter MODE).

Beispiel:

```
10 MODE 2
20 PLOT 50,70
30 MOVE 100,100
40 DRAWR 50,50
```

Anmerkung zur Kompatibilität: Diesen Grafik-Befehl darf man nicht mit dem MOVE-Kommando des Sinclair Spectrum verwechseln, der zur Übertragung von Daten von einem Feld zum anderen dient.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	LOCATE Zahl1,Zahl2
COMMODORE 128	:	LOCATE Zahl1,Zahl2
ENTERPRISE 64,128	:	PLOT Zahl1,Zahl2
MSX-STANDARD	:	PRESET Zahl1,Zahl2
› SCHNEIDER CPC464	:	MOVE Zahl1,Zahl2
› SCHNEIDER CPC664,CPC6128	:	MOVE Zahl1,Zahl2
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

MOVER

Befehlsart: Kommando

MOVER **Zahl 1,Zahl 2**

Beschreibung: Mit diesem Befehl läßt sich der Grafikcursor bewegen, ohne daß ein Punkt, eine Linie oder andere sichtbare Zeichen auf den Bildschirm gebracht werden. Dabei gibt **Zahl 1** die Verschiebung auf der x-Achse, **Zahl 2** die Verschiebung auf der y-Achse wieder.

Der Befehl

```
MOVER 2Ø,3Ø
```

bewegt den Grafikcursor von Punkt a (alte Position) zu Punkt n (neue Position).

Will man den Befehl MOVER einsetzen, muß vorher der Grafikbildschirm aufgerufen werden. Das geschieht mit MODE 2 (siehe MODE).

Beispiel:

```
10 MODE 2
20 PLOT 50,70
30 MOVER 100,100
40 DRAWR 50,50
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :  PRESET STEP Zahl1,Zahl2
> SCHNEIDER CPC464          :  MOVER Zahl1,Zahl2
> SCHNEIDER CPC664,CPC6128  :  MOVER Zahl1,Zahl2
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

MOVSPR

Befehlsart: Anweisung

MOVSPR **Zahl 1,Zahl 2,Zahl 3**
MOVSPR **Zahl 1,Zahl 4#Zahl 5**

Beschreibung: Durch den MOVSPR-Befehl bewegt man ein Sprite mit der durch **Zahl 1** bezeichneten Nummer an eine Stelle, die durch **Zahl 2** (x-Richtung) und **Zahl 3** (y-Richtung) vorgegeben ist. Werden **Zahl 2** und **Zahl 3** als absolute Zahlen eingegeben, so handelt es sich um die absoluten Koordinaten. Durch Angabe eines Vorzeichens(/-) wird die Bewegung im Verhältnis zur letzten Stellung des Grafik-Cursors ausgeführt.

Im zweiten Format des Befehls gibt man an Stelle der Bewegungskoordinaten den Winkel an, um den sich das Sprite dreht. **Zahl 5** bestimmt die Geschwindigkeit der Drehung.

Beispiel:

```
1000 MOVSPR 1,240,0:REM SPRITE 1 BEWEGUNG AUF POS (240,0)
1010 MOVSPR 2,180#7:REM SPRITE 2 BEWEGUNG VON OBEN NACH UNTEN
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  MOVSPR Zahl1,Zahl2,Zahl3
                               MOVSPR Zahl1,+/-Zahl2,+/-Zahl3
                               MOVSPR Zahl1,Zahl2#Zahl3
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
          PUT SPRITE Zahl4,Zahl2,Zahl3,Zahl5,Zahl1
          PUT SPRITE STEP Zahl4,Zahl2,Zahl3,Zahl5,Zahl1
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

NEW

Befehlsart: Kommando

NEW

Beschreibung: Das Kommando NEW löscht das Programm im Speicher des Computers. Zur gleichen Zeit werden auch alle Variablen gelöscht.

NEW läßt sich auch innerhalb eines Programmes verwenden, falls man will, daß der Computer nach Ausführung des Programms frei für neue Programme ist.

Beispiel:

```
10 PRINT "WIE HEISST DU"
20 INPUT NA$
30 PRINT "GUTEN TAG, ";NA$
40 PRINT:PRINT"GUTES COMPUTERN."
50 NEW
```

Anmerkung: Mit NEW wird das Programm nicht im materiellen Sinne "gelöscht". Was gelöscht wird, ist der Zeiger, der auf den Anfang des Programmes weist.

Es gibt einige Programmiertricks, um das mit NEW "gelöschte" Programm zurückzuholen.

Unter anderem gibt es bei einigen BASIC-Erweiterungen

das Kommando OLD. Es macht das Programm wieder benutzbar, solange nach der Eingabe von NEW nicht neue Programme geschrieben oder geladen worden sind.

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  NEW
› COMMODORE VC20,64         :  NEW
› COMMODORE 16,116,PLUS4    :  NEW
› COMMODORE 128             :  NEW
› ENTERPRISE 64,128         :  NEW
› MSX-STANDARD              :  NEW
› SCHNEIDER CPC464          :  NEW
› SCHNEIDER CPC664,CPC6128  :  NEW
› SINCLAIR ZX81             :  NEW
› SINCLAIR SPECTRUM         :  NEW
```

NEW ALL

Befehlsart: Kommando

NEW ALL

Beschreibung: Beim ENTERPRISE können mehrere Programme gleichzeitig im Speicher geladen sein. Mit NEW ALL werden alle Programme gelöscht. Der Befehl ist also dem NEW-Kommando (siehe dort) verwandt, das allerdings nur jeweils ein Programm löscht.

Der Programmzeiger wird nach Ausführung des Befehls automatisch auf Null zurückgestellt.

Beispiel:

NEW ALL

Anmerkung: Das Kommando NEW ALL kann auch innerhalb eines Programmes verwendet werden.

Beispiel:

```
58Ø  INPUT "IST EINES DER GELADENEN PROGRAMME NOCH ZU VER-
     WENDEN";AN$
59Ø  IF AN$="NEIN" THEN NEW ALL
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	NEW ALL
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

NEXT

Befehlsart: Anweisung

FOR...TO...STEP
NEXT **Variablenname 1,Variablenname 2...**

Beschreibung: Die NEXT-Anweisung wird in Verbindung mit der FOR...TO-Anweisung gebraucht (siehe unter FOR...TO... STEP). Wird der NEXT-Befehl vom Programm erreicht, so kehrt es zur FOR-Anweisung zurück, überprüft die Anzahl der abgearbeiteten Schleifen und führt, falls die angegebene Anzahl der Durchläufe noch nicht erreicht ist, die in der Schleife vorgesehenen Programmanweisungen durch. Schreibt man die Anweisung ohne **Variablennamen**, führt das NEXT auf die letzte FOR-Anweisung zurück.

Bei Angabe von mehreren **Variablennamen** werden diese der Reihe nach bearbeitet.

Beispiel:

```
10 FOR A= 1000 TO 1 STEP -10
20 FOR B= 1 TO 1000 STEP 20
30 FOR C= 1 TO 400: NEXT
40 PRINT "*****":PRINT:PRINT A;" + ";B;" = ":PRINT:PRINT
50 PRINT A+B
60 NEXT B,A

READY.
```

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  NEXT Variablenname
> COMMODORE VC20,64         :  NEXT Variablenname
> COMMODORE 16,116,PLUS4    :  NEXT Variablenname
> COMMODORE 128             :  NEXT Variablenname
> ENTERPRISE 64,128         :  NEXT Variablenname
> MSX-STANDARD              :  NEXT Variablenname
> SCHNEIDER CPC464          :  NEXT Variablenname
> SCHNEIDER CPC664,CPC6128  :  NEXT Variablenname
> SINCLAIR ZX81             :  NEXT Variablenname
> SINCLAIR SPECTRUM         :  NEXT Variablenname
```

NOT

Befehlsart: Operator

NOT **Argument**

Beschreibung: Mit NOT lassen sich bei bedingten Verzweigungen bestimmte Argumente ausschließen. Eine Anweisung wird ausgeführt, wenn eine bestimmte Bedingung nicht erfüllt ist.

Beispiel:

siehe Seite 273

NOT erzielt eine "wahr"-Meldung, wenn das Argument "nicht wahr" ist und umgekehrt:

NOT	Argument	Ergebnis
	nicht wahr (Bedingung nicht erfüllt)	wahr (Bedingung erfüllt)
	wahr (Bedingung erfüllt)	nicht wahr (Bedingung nicht erfüllt)

Das Argument kann nicht nur aus Zahlen-Operationen, sondern auch aus Textausdrücken bestehen.

Beispiel:

siehe Seite 273

Intern bearbeitet der Computer eine logische Operation mit NOT, indem er das Argument (den Operator) in eine ganze 16-Bit-Binärzahl umwandelt. Bit für Bit schließt er sie aus, indem er sie gleichsam "umpolt".

Als Beispiel:

```
  PRINT NOT 14
```

ergibt

```
  -15
```

weil

```
                          14 = ØØØØØØØØØØØØ111Ø

             NOT

Ergebnis:                -15 = 1111111111111ØØØ1
```

In eine logische Operation übersetzt bedeutet dies:

NOT

nicht wahr (Ø)	wahr (1)
nicht wahr (Ø)	wahr (1)
nicht wahr (Ø)	wahr (1)
nicht wahr (Ø)	wahr (1)
nicht wahr (Ø)	wahr (1)
nicht wahr (Ø)	wahr (1)
nicht wahr (Ø)	wahr (1)
nicht wahr (Ø)	wahr (1)
nicht wahr (Ø)	wahr (1)
nicht wahr (Ø)	wahr (1)
nicht wahr (Ø)	wahr (1)
nicht wahr (Ø)	wahr (1)
wahr (1)	nicht wahr (Ø)
wahr (1)	nicht wahr (Ø)
wahr (1)	nicht wahr (Ø)
nicht wahr (Ø)	wahr (1)

Wie die anderen logischen Operatoren (siehe unter AND, OR) dient NOT zu Vergleichsoperationen, aber auch zur Bitmanipulation und zur Durchführung Boolescher Operationen. NOT ist daher nicht dasselbe wie ≠.

```
PRINT NOT 1
-2
READY.

PRINT <>1
?SYNTAX  ERROR
READY.
```

Beispiel:

siehe Seite 273

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  NOT Argument
› COMMODORE VC20,64         :  NOT Argument
› COMMODORE 16,116,PLUS4    :  NOT Argument
› COMMODORE 128             :  NOT Argument
› ENTERPRISE 64,128         :  NOT Argument
› MSX-STANDARD              :  NOT Argument
› SCHNEIDER CPC464          :  NOT Argument
› SCHNEIDER CPC664,CPC6128  :  NOT Argument
› SINCLAIR ZX81             :  NOT Argument
› SINCLAIR SPECTRUM         :  NOT Argument
```

```
10 PRINT"■ EINE TEILUNGSAUFGABE"
20 INPUT "GIB DEN ZAEHLER EIN: ";A
30 INPUT "GIB DEN NENNER  EIN: ";B
40 IF NOT B=0 THEN PRINT A;" GETEILT DURCH ";B;" IST ";A/B;".":GOTO 20
50 PRINT "EINE ZAHL DARF MAN NIEMALS DURCH NULL   TEILEN!"

READY.
```

```
10 PRINT "WOLLEN SIE, DASS ICH FUER SIE EIN"
20 INPUT "PROGRAMM LADE ";A$
30 IF NOT A$="JA"THEN PRINT "■SCHADE! DANN STUERZE ICH  ":GOTO 45
40 INPUT "BITTE DEN NAMEN DES PROGRAMMES: ";P$
45 FOR T=1 TO 600:NEXT
50 K$=" "
60 FOR K=1 TO 52:L$=L$+K$
70 PRINTL$;"A":NEXT K
80 PRINT "□           AABBB!"
90 GOTO 90

READY.
```

```
10 INPUT "TIPPEN SIE EINE ZAHL";A
15 IF A=1 THEN PRINT"RICHTIG!":END
20 IF NOT A>10 AND NOT A<-5 THEN PRINT "NAHE DRAN!":GOTO 10
30 IF NOT A>20 AND NOT A<-10 THEN PRINT "FEHLT NOCH EIN STUECK!":GOTO 10
40 IF A>20 OR A<-5 THEN PRINT "WEIT ENTFERNT!":GOTO 10

READY.
```

NOTE

Befehlsart: Kommando

NOTE #**Kanal**, **Sektor**, **Byte**

Beschreibung: Bei Ausführung dieses Befehls wird festgestellt, an welcher Stelle einer bestimmten Diskettendatei der Zeiger dieser Datei steht. Der **Kanal** muß für eine Diskettendatei offen sein. Der Variablen **Sektor** wird die Nummer des Sektors und der Variablen **Byte** die Nummer des entsprechenden Bytes innerhalb dieses Sektors übergeben. Statt Diskettendatei benützt man in diesem Zusammenhang meistens den Ausdruck "Diskettenfile".

Beispiel:

```
10 OPEN #4,4,0,"D:VERSUCH.BAS"
20 NOTE #4,Sekt,Byte
30 PRINT "Sektor Nr.: ";Sekt,"Byte Nr.: ";Byte
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	NOTE #Kanal,Sektor,Byte
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

NUMERIC

Befehlsart: Anweisung

NUMERIC **Variablenliste**

Beschreibung: Die Variablen in der **Variablenliste** werden als Zahlenvariablen festgelegt. Ähnlich dem Befehl DIM (siehe dort) lassen sich mit NUMERIC auch Felder definieren.

Definition von Zahlenvariablen:

NUMERIC **Variablenname**

legt mit dem **Variablennamen** eine Variable als Zahlenvariable fest.

Definition eines Zahlenfeldes: Die möglichen Formate für diesen Befehl sind

NUMERIC **Variablenname(Zahl 2,Zahl 4...)**

oder

NUMERIC **Variablenname(Zahl 1 TO Zahl 2,Zahl 3 TO Zahl 4...)**

Bei der Definition eines Zahlenfeldes gibt der **Variablenname** die Namen der indizierten Variablen an. **Zahl 2,Zahl 4...** usw. geben die Obergrenze eines Feldes an. **Zahl 1, Zahl 3...** usw. geben die Untergrenze eines Feldes an. Normalerweise hat diese Zahl den Wert Ø.

Anmerkung: Die Befehlsformate lassen sich auch kombinieren, wie aus dem Beispiel ersichtlich.

Beispiel:

```
NUMERIC APFEL,BIRNE(-3TO12,13)
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  NUMERIC Variablenliste
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

OCT$

Befehlsart: Funktion

OCT$(**Zahl**)

Beschreibung: Dieser Befehl wandelt eine dezimale **Zahl** in eine oktale Zahl um. Sie wird als String wiedergegeben.

Wenn man eine Komma-Zahl eingibt, wird diese automatisch gerundet.

Beispiel:

```
10 FOR i=-32768 TO 65535
20 PRINT CHR$(30);OCT$(i)
30 NEXT i
```

Anmerkung: Das oktale Zahlensystem basiert nur auf den Ziffern Ø bis 7.

Beim dezimalen Zahlensystem zählen wir: Ø,1,2,3,4,5,6,7,8,9; nun kommt die Zahl 1Ø. Dabei schaltet - bildlich gesprochen - die 9 auf die Ø zurück und zählt die nächste Stelle um eins hoch. Nun wird weitergezählt: 11,12,13...

Beim oktalen Zahlensystem zählt man: Ø,1,2,3,4,5,6,7;

nun kommt die Ziffernkombination 1Ø. Sie entspricht der Zahl 8 in unserem Zehnersystem. Dabei schaltet die 7 auf die Ø zurück und zählt die nächste Stelle um eins hoch. Nun wird weitergezählt: 11,12,13...; dabei sagen wir nicht "zehn, elf, zwölf, dreizehn", sondern eins-null oktal, eins-eins oktal, eins-zwei oktal, eins-drei oktal .... usw.

Die 2Ø ersten Zahlen des Dezimalsystems im Vergleich mit den ersten 2Ø Zahlen des Oktalsystems:

dezimales Zahlensystem:
Ø 1 2 3 4 5 6 7 8 9 1Ø 11 12 13 14 15 16 17 18 19 2Ø

oktales Zahlensystem:
Ø 1 2 3 4 5 6 7 1Ø 11 12 13 14 15 16 17 2Ø 21 22 23 24

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
› MSX-STANDARD              :  OCT$(Zahl)
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

ON BREAK GOSUB

Befehlsart: Anweisung

ON BREAK GOSUB **Zeilennummer**

Sobald die ESC-Taste zweimal gedrückt worden ist, erfolgt der Sprung in ein Unterprogramm, das mit der **Zeilennummer** beginnt. Das Programm wird also nicht, wie es ohne diesen Befehl der Fall wäre, abgebrochen.

Der Befehl läßt sich mit ON BREAK STOP (siehe dort) wieder aufheben.

Beispiel:

```
10 ON BREAK GOSUB 100
20 PRINT "Endlosschleife":GOTO 20
30 END
100 PRINT:PRINT:PRINT"Pause":PRINT:PRINT
110 RETURN
```

Anmerkung: Bei unglücklicher Programmierung kann es passieren, daß ein einmal gestartetes Programm mit einem ON BREAK GOSUB-Befehl nicht mehr abgebrochen werden kann, es sei denn durch Totalreset. Falls ein solches Programm vorher nicht abgespeichert wurde, ist es in diesem Fall verloren.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	ON STOP GOSUB Zeilennummer
› SCHNEIDER CPC464	:	ON BREAK GOSUB Zeilennummer
› SCHNEIDER CPC664,CPC6128	:	ON BREAK GOSUB Zeilennummer
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

ON BREAK STOP

Befehlsart: Anweisung

ON BREAK STOP

Beschreibung: Dieser Befehl hebt die Wirkung des Befehls ON BREAK GOSUB (siehe dort) auf.

Schreibt man ON BREAK STOP in jenes Unterprogramm, das mit ON BREAK GOSUB angesprungen wird, läßt sich das Programm nach einmaligem Durchlauf wieder unterbrechen, indem man die ESC-Taste zweimal betätigt.

Beispiel:

```
10 ON BREAK GOSUB 100
20 PRINT "Endlosschleife";:GOTO 20
30 END
100 PRINT:PRINT:PRINT"Pause":PRINT:PRINT
110 ON BREAK STOP
120 RETURN
```

Kompatibilitätsliste:

	ATARI 600,800,130XE	:
	COMMODORE VC20,64	:
	COMMODORE 16,116,PLUS4	:
	COMMODORE 128	:
	ENTERPRISE 64,128	:
	MSX-STANDARD	:
›	SCHNEIDER CPC464	: ON BREAK STOP
›	SCHNEIDER CPC664,CPC6128	: ON BREAK STOP
	SINCLAIR ZX81	:
	SINCLAIR SPECTRUM	:

ON ERROR GOTO

Befehlsart: Anweisung

ON ERROR GOTO **Zeilennummer**

Beschreibung: Tritt ein Fehler auf, wird das Programm nicht wie sonst üblich abgebrochen, sondern es wird zu der Programmzeile gesprungen, die durch die **Zeilennummer** gekennzeichnet ist.

In dieser angesprungenen Zeile läßt sich eine Korrektur oder eine Meldung unterbringen. Eine Hilfe hierzu bieten auch die Befehle ERL und ERR (siehe dort).

Mit der Anweisung RESUME (siehe dort) kann man in das Programm zurückspringen.

Beispiel:

```
10 ON ERROR GOTO 100
20 PRINT "Name"
30 GOTO 1000
40 END
100 PRINT "Es ist ein Fehler im Programm!"
110 RESUME 40
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :  TRAP Zeilennummer
  COMMODORE 128             :  TRAP Zeilennummer
  ENTERPRISE 64,128         :  WHEN EXCEPTION USE Handlername
> MSX-STANDARD              :  ON ERROR GOTO Zeilennummer
> SCHNEIDER CPC464          :  ON ERROR GOTO Zeilennummer
> SCHNEIDER CPC664,CPC6128  :  ON ERROR GOTO Zeilennummer
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

ON...GOSUB

Befehlsart: Anweisung

ON **Variablenname** GOSUB **Zeilennummer**
ON **Variablenname** GOSUB **Zeilennummer 1,Zeilennummer 2,Zeilennummer 3,...**

Beschreibung: Der Befehl wirkt ähnlich der Anweisung ON...GOTO (siehe dort). Allerdings definiert man mit den **Zeilennummern** jeweils den Beginn eines Unterprogrammes, von dem das Programm abhängig von jenem Zahlenwert abgearbeitet wird, der für den **Variablennamen** eingegeben wurde.

Beispiel:

```
10 INPUT Y
20 ON Y GOSUB 100,200,300
30 IF Y>3 OR Y<1 THEN PRINT "FALSCHE EINGABE"
40 GOTO 10
100 PRINT"    ┌──────┐
110 PRINT"    │      │
120 PRINT"    └──────┘
125 PRINT:PRINT
130 RETURN
200 PRINT "   ┌┐
210 PRINT "   ││
220 PRINT "   └┘
225 PRINT:PRINT
230 RETURN
300 PRINT "   ┌─────/
310 PRINT "   └─────\
315 PRINT:PRINT
320 RETURN

READY.
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	ON Variable GOSUB Zeilennummern
› COMMODORE VC20,64	:	ON Variable GOSUB Zeilennummern
› COMMODORE 16,116,PLUS4	:	ON Variable GOSUB Zeilennummern
› COMMODORE 128	:	ON Variable GOSUB Zeilennummern
› ENTERPRISE 64,128	:	ON Variable GOSUB Zeilennummern
› MSX-STANDARD	:	ON Variable GOSUB Zeilennummern
› SCHNEIDER CPC464	:	ON Variable GOSUB Zeilennummern
› SCHNEIDER CPC664,CPC6128	:	ON Variable GOSUB Zeilennummern
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

ON...GOTO

Befehlsart: Anweisung

ON **Variablenname** GOTO **Zeilennummer**
ON **Variablenname** GOTO **Zeilennummer 1,Zeilennummer 2,Zeilennummer 3...**

Beschreibung: Bei Eingabe einer Variablen über INPUT (siehe dort) springt das Programm in eine bestimmte **Zeilennummer** oder in mehrere **Zeilennummern**. Die für den **Variablennamen** eingegebene Zahl bestimmt, in welche der angegebenen **Zeilennummern** das Programm springt. Gibt man eine 1, so springt das Programm in die **Zeilennummer 1** und arbeitet die folgenden Zeilennummern ab, sofern es durch keine END-Anweisung (siehe dort) oder einen Sprungbefehl daran gehindert wird. Bei Eingabe einer 2 wird zur Zeile mit **Zeilennummer 2** gesprungen, usw.

Sobald man bei diesem Format Ø eingibt oder eine negative Zahl bzw. einen Wert größer als die Anzahl der angegebenen **Zeilennummern** verwendet, wird das Programm mit jenem Befehl weiter bearbeitet, der auf die ON...GOTO-Anweisung folgt.

Beispiele:

```
10 INPUT X
20 ON X GOTO 40,50,60
30 IF X>3 OR X<1 THEN PRINT "FALSCHE EINGABE":GOTO 10
40 PRINT "ES WURDE DIE >EINS< GEDRUECKT":END
50 PRINT "ES WURDE DIE >ZWEI< GEDRUECKT":END
60 PRINT "ES WURDE DIE >DREI< GEDRUECKT":END

READY.
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	ON Variable GOTO Zeilennummern
› COMMODORE VC20,64	:	ON Variable GOTO Zeilennummern
› COMMODORE 16,116,PLUS4	:	ON Variable GOTO Zeilennummern
› COMMODORE 128	:	ON Variable GOTO Zeilennummern
› ENTERPRISE 64,128	:	ON Variable GOTO Zeilennummern
› MSX-STANDARD	:	ON Variable GOTO Zeilennummern
› SCHNEIDER CPC464	:	ON Variable GOTO Zeilennummern
› SCHNEIDER CPC664,CPC6128	:	ON Variable GOTO Zeilennummern
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

ON INTERVAL GOSUB

Befehlsart: Anweisung

ON INTERVAL=**Zahl** GOSUB **Zeilennummer**

Beschreibung: Mit diesem Befehl definieren wir den Sprung zu einem Unterprogramm. Dieser Sprung wird bei eingeschalteter Uhr nach bestimmten Zeitabständen ausgeführt. Das normale Programm wird dann unterbrochen, und das Unterprogramm wird abgearbeitet. Danach erfolgt der Rücksprung in das normale Programm.

Die **Zahl** gibt die Zeitspanne an, nach der in das Unterprogramm gesprungen wird. Sie ergibt sich aus der Formel

Zeitspanne= **Zahl** * Ø.Ø2 Sekunden.

Die **Zahl** kann zwischen 1 und 65535 liegen. Erlaubt sind auch die **Zahl**en von -32768 bis -1; sie entsprechen allerdings den Werten 32768 bis 65535. Ihre Eingabe ist also nicht besonders sinnvoll.

Die **Zeilennummer** gibt die Programmzeile an, mit der das Unterprogramm beginnt. Es muß mit RETURN (siehe dort) abgeschlossen werden (siehe auch GOSUB...RETURN).

ON INTERVAL GOSUB

Beispiel:

```
10 ON INTERVAL=50 GOSUB 100
20 INTERVAL ON
30 PRINT "DA DA DA"
40 GOTO 30
100 PRINT "AHA AHA"
110 RETURN
```

Anmerkungen: Bei der Anwendung des Befehles sollte man folgendes beachten:

- Der Befehl ON INTERVAL=**Zahl** GOSUB **Zeilennummer** sollte am Anfang eines Programms stehen.
- Diese Unterbrechungsroutine muß durch den Befehl INTERVAL ON (siehe unter INTERVAL ON/OFF/STOP) eingeschaltet werden.
- Mit INTERVAL OFF läßt sich diese Unterbrechungsroutine abschalten.
- Mit INTERVAL STOP werden in der Unterbrechungsroutine die Unterbrechungen gespeichert und nach der Ausführung der Routine abgearbeitet.

Anmerkung zur Kompatibilität: Bei den Schneider-Computern entspricht dieser Anweisung weitgehend der Befehl EVERY **Zahl** GOSUB **Zeilennummer** (siehe EVERY GOSUB).

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	ON INTERVAL=Zahl1 GOSUB Zeilennummer
SCHNEIDER CPC464	:	EVERY Zahl1,Zahl2 GOSUB Zeilennummer
SCHNEIDER CPC664,CPC6128	:	EVERY Zahl1,Zahl2 GOSUB Zeilennummer
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

ON KEY GOSUB

Befehlsart: Anweisung

ON KEY GOSUB **Zeilennummer**

Beschreibung: Nach Drücken einer vorher festgelegten Funktionstaste (KEY) verzweigt das Programm in eine mit der **Zeilennummer** beginnende Subroutine. Wie beim ON GOSUB-Befehl (siehe ON GOSUB) können mehrere **Zeilennummern** auf die GOSUB-Anweisung folgen. Sie werden durch Kommata getrennt. Die Subroutinen werden dann in der angegebenen Reihenfolge abgearbeitet. Die erste **Zeilennummer** wird dem Drücken der Funktionstaste 1 zugeordnet, die zweite der Funktionstaste 2 usw.

Will man bestimmte Funktionstasten in diesem Format nicht verwenden, so wird die entsprechende **Zeilennummer** weggelassen, d. h., in diesem Falle folgen zwei Kommata aufeinander.

Beispiel:

```
ON KEY GOSUB 231Ø,4ØØØ,,5ØØØ
```

Es müssen so viele **Zeilennummern** eingegeben werden, wie die Zahl der höchsten verwendeten Funktionstasten beträgt. Bei Verwendung der Tasten F1, F2, F6, F7 hat die Anweisung also folgendes Format:

ON KEY GOSUB **Zeilennummer 1,Zeilennummer 2,,,Zeilennummer 3,Zeilennummer 4**

Beispiel:

```
10 ONKEYGOSUB100,,,,200
20 KEY1,"":KEY2,"PUNKTE":KEY3,"":KEY4,"":KEY5,"PUNKTE"
30 KEY(1)ON:KEY(5)ON
40 X=INT(RND(1)*1.03)*10+32
50 IFX=42 THENP=10
60 PRINTCHR$(X);
70 P=P-1
80 GOTO40
100 F1=F1+P:IFF1<0 THENF1=0
110 IFF1>999 THENEND
120 KEY1,"F1 "+RIGHT$(" "+STR$(INT(F1)),3)
130 RETURN
200 F5=F5+P:IFF5<0 THENF5=0
210 IFF5>999 THENEND
220 KEY5,"F5 "+RIGHT$(" "+STR$(INT(F5)),3)
230 RETURN
```

Das Beispielprogramm ist ein kleines Spiel. Sobald ein Sternchen auf dem Bildschirm erscheint, müssen die bis zu zwei Spieler ihre jeweilige Funktionstaste drücken, Spieler 1 F1, Spieler 2 F5.

In der untersten Zeile werden die Punkte angezeigt. Dies veranschaulicht auch den Gebrauch des KEY-Befehls (siehe KEY).

Drückt man eine der beiden Funktionstasten F1 oder F5, wird in die sogenannten Unterbrechungsroutinen verzweigt, in der die Punktestände festgelegt werden.

Anmerkung: Der Befehl sollte am Anfang eines Programms stehen. Danach muß man mit KEY(x)ON (siehe KEY(x)ON/OFF/STOP) die Unterbrechungsfunktion einschalten.

Mit KEY(x)OFF wird sie wieder abgeschaltet.

KEY(x)STOP verhindert eine Unterbrechung in der Unterbrechungsroutine.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	ON KEY GOSUB Zeilennummern
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

ON SPRITE GOSUB

Befehlsart: Anweisung

ON SPRITE GOSUB **Zeilennummer**

Beschreibung: Mit dem Befehl wird bei einer Spritekollision der normale Programmablauf unterbrochen. Das Programm verzweigt in das Unterprogramm, das mit Programmzeile **Zeilennummer** beginnt.

Mit RETURN kehrt man wieder an die Stelle im Programm zurück, an der es unterbrochen wurde.

Beispiel:

```
ON SPRITE GOSUB 1000
```

Anmerkung: Dieser Befehl sollte am Anfang eines Programmes stehen.

Mit SPRITE ON (siehe SPRITE ON/OFF/STOP) schaltet man die Kollisionsabfrage der Sprites ein.

Mit SPRITE OFF läßt sich dieser Modus wieder abschalten.

Mit SPRITE STOP verhindert man eine Unterbrechung in der Subroutine. Nach Beendigung dieser Routine wird sie nochmals ausgeführt.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	COLLISION Zahl,Zeilennummer
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	ON SPRITE GOSUB Zeilennummer
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

ON SQ GOSUB

Befehlsart: Anweisung

ON SQ (**Zahl**) GOSUB **Zeilennummer**

Beschreibung: Dieser Befehl überprüft, ob in einer Tonwarteschlange des Kanals **Zahl** ein Platz frei ist. In diesem Fall wird in das Unterprogramm gesprungen, das mit der **Zeilennummer** beginnt.

Man kann die 3 Tonkanäle des CPC überprüfen. Dabei entspricht:

Kanal A **Zahl** = 1
Kanal B **Zahl** = 2
Kanal C **Zahl** = 4

Gibt man andere Werte ein, erhält man eine Fehlermeldung. Das Unterprogramm muß mit einem RETURN abgeschlossen werden (siehe auch GOSUB).

Beispiel:

```
ON SQ (2) GOSUB 2000
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	ON SQ(Zahl) GOSUB Zeilennummer
› SCHNEIDER CPC664,CPC6128	:	ON SQ(Zahl) GOSUB Zeilennummer
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

ON STOP GOSUB

Befehlsart: Anweisung

ON STOP GOSUB **Zeilennummer**

Beschreibung: Mit dieser Anweisung wird der Abbruch eines BASIC-Programmes durch Drücken von CTRL und STOP verhindert. Dabei wird der normale Programmablauf unterbrochen, und es wird in die Routine abgezweigt, die in **Zeilennummer** beginnt.

ON STOP GOSUB

Mit diesem Befehl kann eine "unerlaubte" Unterbrechung durch Benutzer des Programmes verhindert werden.

Mit dem Befehl RETURN (siehe RETURN) wird von dem angesprungenen Unterprogramm wieder an die Stelle des Programms zurückgekehrt, wo das Programm unterbrochen wurde.

Beispiel:

```
10 ONSTOPGOSUB100
20 STOPON
30 COLOR(timeAND15),(timeAND15),(timeAnd15)
40 GOTO30
50 IF INKEY$=""THEN30
60 END
100 CLS
110 PRINT"so nicht!"
120 FOR1=0 TO500
130 RETURN50
```

Anmerkung: Der Befehl sollte stets am Anfang eines Programmes eingesetzt werden.

Mit STOP ON (siehe STOP ON/OFF/STOP) wird ein Programmabbruch verhindert.

Nach dem Befehl STOP OFF kann man das Programm wieder abbrechen.

Mit STOP STOP wird die Unterbrechung so lange verzögert, bis eine Routine abgearbeitet ist.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	ON STOP GOSUB Zeilennummer
SCHNEIDER CPC464	:	ON BREAK GOSUB Zeilennummer
SCHNEIDER CPC664,CPC6128	:	ON BREAK GOSUB Zeilennummer
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

ON STRIG GOSUB

Befehlsart: Anweisung

ON STRIG GOSUB **Zeilennummer 1,Zeilennummer 2...**

Beschreibung: Beim Drücken der Leertaste oder eines Joystick-Feuerknopfes wird in ein Unterprogramm gesprungen, das durch **Zeilennummer 1** oder weitere **Zeilennummern** angegeben ist. **Zeilennummer 1** gilt für die Leertaste, **Zeilennummer 2** für Joystick 1, Auslöser 1 und so weiter.

Beispiel:

```
10 ON STRIG GOSUB 100,200
20 STRIG(0) ON:STRIG(1) ON
30 GOTO 30
100 PRINT " LEERTASTE ": RETURN
200 PRINT " JOYSTICK ": RETURN
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	ON STRIG GOSUB Zeilennummern
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

OPEN

Befehlsart: Anweisung

OPEN **Parameterliste**

für Commodore-Rechner:

OPEN **Kanalnummer,Zahl 1,Zahl 2**

Beschreibung: Der Datenkanal mit der **Kanalnummer** wird eröffnet. Hierdurch kann der Benutzer auf Informationen von Peripherie-Geräten wie Kassettenrecorder, Disketten-Laufwerk, Drucker usw. zugreifen bzw. Informationen ausgeben. Der Wert für die **Kanalnummer** ist ganzzahlig und positiv. Seine Obergrenze variiert von Computertyp zu Computertyp.

Da das Format des Befehls von Computertyp zu Computertyp verschieden ist, wurde dieser Befehl für die Commodore-Rechner erklärt. Bei den anderen Rechnertypen wird empfohlen, im jeweiligen Handbuch nachzuschauen.

Zahl 1 gibt die Geräteadresse an. Dabei kann zum Beispiel der Wert Ø den Bildschirm ansprechen, der Wert 4 den Drucker und der Wert 8 das Disketten-Laufwerk. Der Wert dieser Zahl ist nicht nur von Computer zu Computer verschieden, er läßt sich auch softwareseitig einstellen.

Zahl 2 gibt an, welche Operation durchgeführt werden soll:

Zahl 2

Ø	Der Datenträger ist zu lesen
1	Der Datenträger ist zu beschreiben
2	Bei Programmende wird ein Bandende-Kennzeichen gesetzt.

Beispiele:

```
OPEN 1,4
```

eröffnet den Kanal 1 über den Drucker

```
OPEN 4,8,15
```

eröffnet den Datenkanal 4, der die Verbindung zwischen Disketten-Laufwerk und Rechner herstellt.

Anmerkung: Die OPEN-Anweisung läßt sich sowohl im Direktmodus als auch innerhalb eines Programmes verwenden.

Anmerkung zur Kompatibilität: Das Format des Befehls ist von Computertyp zu Computertyp verschieden.

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  OPEN #Dateinr,Parameterliste
> COMMODORE VC20,64         :  OPEN Dateinr,Parameterliste
> COMMODORE 16,116,PLUS4    :  OPEN Dateinr,Parameterliste
> COMMODORE 128             :  OPEN Dateinr,Parameterliste
> ENTERPRISE 64,128         :  OPEN #Dateinr,Parameterliste
> MSX-STANDARD              :  OPEN Parameterliste
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
> SINCLAIR SPECTRUM         :  OPEN #Dateinr,Parameterliste
```

OPENIN

Befehlsart: Anweisung

OPENIN **Dateiname**

Beschreibung: Der Befehl eröffnet eine Eingabedatei mit dem **Dateinamen.**

Beispiel:

```
10 REM Die Datei >Woertersee< wird geladen
20 OPENIN "Woertersee"
30 CLOSEIN:PRINT "Woertersee ist geladen"
```

Anmerkung: Beim Einlesen von Kassette kann mit einem Ausrufezeichen (!) vor dem **Dateinamen** verhindert werden, daß die gewohnten Meldungen auf dem Bildschirm erscheinen.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :  OPEN #Zahl,Parameterliste
  COMMODORE VC20,64         :  OPEN Zahl,Parameterliste
  COMMODORE 16,116,PLUS4    :  OPEN Zahl,Parameterliste
  COMMODORE 128             :  OPEN Zahl,Parameterliste
  ENTERPRISE 64,128         :  OPEN #Zahl,Parameterliste
  MSX-STANDARD              :  OPEN Parameterliste
> SCHNEIDER CPC464          :  OPENIN "Dateiname"
> SCHNEIDER CPC664,CPC6128  :  OPENIN "Dateiname"
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :  OPEN #Zahl,Parameterliste
```

OPENOUT

Befehlsart: Anweisung

OPENOUT **Dateiname**

Beschreibung: Der Befehl eröffnet eine Ausgabedatei mit dem **Dateinamen**. (Siehe auch unter OPENIN, CLOSEOUT.)

Beispiel:

```
1000 OPENOUT "Woertersee"
1010 FOR g= 1 TO 20
1020 INPUT "Gib das naechste Wort  jetzt ein:";A$
1030 WRITE#9,A$
1040 NEXT g
1050 CLOSEOUT:PRINT "Woertersee ist gesichert!"
```

Anmerkung: Bei einer Ausgabe über Kassette läßt sich durch ein Ausrufezeichen (!) vor dem **Dateinamen** verhindern, daß die gewohnten Meldungen auf dem Bildschirm erscheinen.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE          :  OPEN #Zahl,Parameterliste
  COMMODORE VC20,64            :  OPEN Zahl,Parameterliste
  COMMODORE 16,116,PLUS4       :  OPEN Zahl,Parameterliste
  COMMODORE 128                :  OPEN Zahl,Parameterliste
  ENTERPRISE 64,128            :  OPEN #Zahl,Parameterliste
  MSX-STANDARD                 :  OPEN Parameterliste
> SCHNEIDER CPC464             :  OPENOUT "Dateiname"
> SCHNEIDER CPC664,CPC6128     :  OPENOUT "Dateiname"
  SINCLAIR ZX81                :
  SINCLAIR SPECTRUM            :  OPEN #Zahl,Parameterliste
```

OPTION ANGLE

Befehlsart: Anweisung

OPTION ANGLE DEGREES
OPTION ANGLE RADIANS

Beschreibung: Hiermit wird die Maßeinheit für die Winkelfunktionen festgelegt. Sie werden in Grad (Degrees) oder im Winkelmaß (Radians) angegeben.

Mit OPTION ANGLE DEGREES (Option Gradmaß) kann man Bogenmaß auf das Gradmaß umschalten. Mit OPTION ANGLE RADIANS wird vom Gradmaß wieder auf das Bogenmaß zurückgeschaltet. Beim Einschalten befindet sich der Rechner im Bogenmaß-Modus.

Beispiel:

```
10 OPTION ANGLE DEGREES
20 PRINT "TANGENS-WERTE in Grad"
30 INPUT T
40 PRINT TAN(T)
50 INPUT "WEITER (J/N)";A$
60 IF A$="J" THEN GOTO 20
70 OPTION ANGLE RADIANS
```

Anmerkungen zur Kompatibilität: Eine Reihe von Computern verfügt nur über den Bogenmaß-Modus.
Bei anderen Rechnern (siehe Kompatibilitätsliste) entspricht

OPTION ANGLE DEGREES dem Begriff DEG

und

OPTION ANGLE RADIANS dem Begriff RAD

Kompatibilitätsliste:

```
ATARI 600,800,130XE       :  DEG
                             RAD
COMMODORE VC20,64         :
COMMODORE 16,116,PLUS4    :
COMMODORE 128             :
› ENTERPRISE 64,128       :  OPTION ANGLE DEGREES
                             OPTION ANGLE RADIANS
MSX-STANDARD              :
SCHNEIDER CPC464          :  DEG
                             RAD
SCHNEIDER CPC664,CPC6128  :  DEG
                             RAD
SINCLAIR ZX81             :
SINCLAIR SPECTRUM         :
```

OR

Befehlsart: Operator

Argument 1 OR **Argument 2**

Beschreibung: Mit OR lassen sich mehrere Bedingungen miteinander logisch verknüpfen. Das heißt: Bei einer bedingten Verzweigung mit IF...THEN (siehe dort) verlangt der logische Operator OR nach der Erfüllung entweder der einen oder der anderen Bedingung.

Beispiel:

```
10 PRINT "GEBEN SIE BITTE EINE ZAHL"
20 INPUT "ZWISCHEN 10 UND 20 EIN:";A
30 IF A<10 OR A>20 THEN PRINT"FALSCHE EINGABE!":GOTO 10
40 PRINT "DIE ZAHL HAT DEN WERT ";A

READY.
```

OR vergleicht die beiden Argumente, um zu einem Ergebnis zu kommen:

Argument 1	OR	Argument 2	Ergebnis
wahr (Bedingung erfüllt)		wahr (Bedingung erfüllt)	wahr (Bedingung erfüllt)
wahr (Bedingung erfüllt)		nicht wahr (Bedingung nicht erfüllt)	wahr (Bedingung erfüllt)
nicht wahr (Bedingung nicht erfüllt)		wahr (Bedingung erfüllt)	wahr (Bedingung erfüllt)
nicht wahr (Bedingung nicht erfüllt)		nicht wahr (Bedingung nicht erfüllt)	nicht wahr (Bedingung nicht erfüllt)

Die Argumente bei einem Mehrfachvergleich mit OR können nicht nur aus Zahlen-Operationen, sondern auch aus Textausdrücken bestehen.

Beispiel:

```
10 PRINT "SIND SIE DER MEINUNG, DASS COMPUTER"
20 INPUT "DENKEN KOENNEN";A$
30 IF A$="JA" OR A$="NEIN" THEN GOTO 60
40 PRINT "SIE MEINEN ALSO:"
50 PRINT A$:END
60 PRINT "SIE MEINEN ALSO, DIE FRAGE KANN MAN"
70 PRINT "SO EINFACH MIT 'JA' ODER 'NEIN' BEANT-  WORTEN?"

READY.
```

OR

Intern bearbeitet der Computer eine logische Operation mit OR, indem er die beiden Argumente (Operatoren) in ganze 16-Bit-Binärzahlen umwandelt. Der Vergleich wird nun Bit für Bit durchgeführt.

```
PRINT 15 OR 14
```

ergibt

```
15
```

```
                          15 = ØØØØØØØØØØØØ1111
          OR
                          14 = ØØØØØØØØØØØØ111Ø

Ergebnis:                      ØØØØØØØØØØØØ1111
```

In eine logische Operation übersetzt bedeutet dieser Vergleich:

	OR		ergibt	
nicht wahr	(Ø)	nicht wahr	(Ø)	nicht wahr (Ø)
nicht wahr	(Ø)	nicht wahr	(Ø)	nicht wahr (Ø)
nicht wahr	(Ø)	nicht wahr	(Ø)	nicht wahr (Ø)
nicht wahr	(Ø)	nicht wahr	(Ø)	nicht wahr (Ø)
nicht wahr	(Ø)	nicht wahr	(Ø)	nicht wahr (Ø)
nicht wahr	(Ø)	nicht wahr	(Ø)	nicht wahr (Ø)
nicht wahr	(Ø)	nicht wahr	(Ø)	nicht wahr (Ø)
nicht wahr	(Ø)	nicht wahr	(Ø)	nicht wahr (Ø)
nicht wahr	(Ø)	nicht wahr	(Ø)	nicht wahr (Ø)
nicht wahr	(Ø)	nicht wahr	(Ø)	nicht wahr (Ø)
nicht wahr	(Ø)	nicht wahr	(Ø)	nicht wahr (Ø)
nicht wahr	(Ø)	nicht wahr	(Ø)	nicht wahr (Ø)
wahr	(1)	wahr	(1)	wahr (1)
wahr	(1)	wahr	(1)	wahr (1)
wahr	(1)	wahr	(1)	wahr (1)
wahr	(1)	nicht wahr	(Ø)	wahr (1)

Wie die anderen logischen Operatoren (siehe AND, NOT) dient auch OR zum Testen von Mehrfachvergleichen, aber auch zur Bit-Manipulation und zur Durchführung Boolescher Operationen.

Beispiel:

```
10 INPUT "□GIB DEINEN GEBURTSTAG EIN: ";A
20 INPUT "□DEN GEBURTSMONAT: ";B
30 INPUT "□DAS JAHR: ";C
35 IF B=12 OR B=1 OR B=2 THEN A$="WINTER"
40 IF B=3 OR B=4 OR B=5 THEN A$="FRUEHLING"
50 IF B=6 OR B=7 OR B=8 THEN A$="SOMMER"
60 IF B=9 OR B=10 OR B=11 THEN A$="HERBST"
70 PRINT "□□□□DER ";A;". ";B". ";C:PRINT
80 PRINT:PRINT "DAS WAR IN EINEM BESONDERS":PRINT"SCHOENEN ";A$

READY.
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	Argument1 OR Argument2
› COMMODORE VC20,64	:	Argument1 OR Argument2
› COMMODORE 16,116,PLUS4	:	Argument1 OR Argument2
› COMMODORE 128	:	Argument1 OR Argument2
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	Argument1 OR Argument2
› SCHNEIDER CPC464	:	Argument1 OR Argument2
› SCHNEIDER CPC664,CPC6128	:	Argument1 OR Argument2
› SINCLAIR ZX81	:	Argument1 OR Argument2
› SINCLAIR SPECTRUM	:	Argument1 OR Argument2

ORD

Befehlsart: Funktion

ORD(**String**)

Beschreibung: Die Funktion gibt für das erste Zeichen des **String** die entsprechende Zahl im ASC-Code zurück.

Beispiel:

```
PRINT ORD("APFEL")
 65
READY
```

Anmerkungen zur Kompatibilität: In BASIC wird im allgemeinen zur Erfüllung dieser Funktion der ASC-Befehl (siehe dort) verwendet. Eine Variante dazu bildet der Befehl CODE (siehe dort).

Die ORD-Funktion ist eine Anleihe aus der Programmiersprache PASCAL.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :  ASC(String)
  COMMODORE VC20,64         :  ASC(String)
  COMMODORE 16,116,PLUS4    :  ASC(String)
  COMMODORE 128             :  ASC(String)
> ENTERPRISE 64,128         :  ORD(String)
  MSX-STANDARD              :  ASC(String)
  SCHNEIDER CPC464          :  ASC(String)
  SCHNEIDER CPC664,CPC6128  :  ASC(String)
  SINCLAIR ZX81             :  CODE String
  SINCLAIR SPECTRUM         :  CODE String
```

ORIGIN

Befehlsart: Kommando

ORIGIN Zahl 1,Zahl 2,Zahl 3,Zahl 4,Zahl 5,Zahl 6

Beschreibung: Mit diesem Kommando lassen sich ein neuer Grafikcursorursprung und ein Grafikfenster festlegen.

Will man nur einen neuen Ursprung festlegen, so ist lediglich der folgende Teil des Kommandos zu verwenden:

ORIGIN Zahl 1,Zahl 2

Die Zahlen **Zahl 1,Zahl 2** geben dabei die neuen Koordinaten des Ursprungs an. Die Zahl **Zahl 1** ist die x-Koordinate und die Zahl **Zahl 2** ist die y-Koordinate. Der dann festgelegte Punkt hat daraufhin die Koordinaten (Ø,Ø) für die folgenden Grafikbefehle: (siehe MOVE, PLOT, DRAW, ...). Falls man den ORIGIN-Befehl anwendet, erhält die untere linke Ecke die Koordinaten (Ø,Ø). Die Koordinaten werden nach rechts, bzw. nach oben hin größer.

Anhand der vier Zahlen **Zahl 3,Zahl 4,Zahl 5,Zahl 6** kann man ein Fenster definieren, in dem die Punkte nur dann gezeichnet werden, wenn diese innerhalb des Fensters liegen. Die Zahlen werden ab der unteren linken Ecke gezählt, und nicht durch einen von ORIGIN definierten Ausgangspunkt. Die **Zahl 3** ist dabei die linke Grenze des Fensterrahmens, die **Zahl 4** die rechte, die **Zahl 5** die obere, und die **Zahl 6** die untere.

Es sind nur Zahlen von Ø-639 sinnvoll.

Beispiel:

Zeile 1Ø	Umschalten auf 8Ø-Zeichenmodus
Zeile 2Ø	Umschalten auf Winkelgradmaß
Zeile 3Ø	Definieren eines neuen Ursprungs
Zeile 7Ø	Definieren eines neuen Ursprungs plus Fenster obere Hälfte des Bildschirms
Zeile 11Ø-18Ø	Unterprogramm für Sinus-Zeichnen. Mit einem Tastendruck gelangt man einen Punkt weiter.

```
10 MODE 2
20 DEG
30 ORIGIN 20,200
40 PRINT "sinus-funktion"
50 GOSUB 110
60 CLS
70 ORIGIN 20,200,0,639,399,200
80 PRINT "obere Teil einer sinus-funktion"
90 GOSUB 110
100 END
110 REM sinus-funktion
120 PLOT -20,0:DRAW 619,0
130 PLOT 0,-200:DRAW 0,180
140 FOR i=-20 TO 619
150 PLOT i,100*SIN(i)
160 NEXT
170 IF INKEY$="" THEN 170
180 RETURN
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :
    ORIGIN Zahl1,Zahl2,Zahl3,Zahl4,Zahl5,Zahl6
> SCHNEIDER CPC664,CPC6128  :
    ORIGIN Zahl1,Zahl2,Zahl3,Zahl4,Zahl5,Zahl6
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

OUT

Befehlsart: Kommando

OUT **Adresse, Byte**

Beschreibung: Mit diesem Befehl kann man ein **Byte**, also eine Zahl zwischen Ø und 255, an den durch die **Adresse** angegebenen Interfacebaustein senden.

Dabei lassen sich dezimale ebenso wie hexadezimale oder binäre Zahlen verwenden.

Beispiel:

OUT &F8ØØ,12

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  OUT Adresse,Zahl
> MSX-STANDARD              :  OUT Adresse,Zahl
> SCHNEIDER CPC464          :  OUT Adresse,Zahl
> SCHNEIDER CPC664,CPC6128  :  OUT Adresse,Zahl
  SINCLAIR ZX81             :
> SINCLAIR SPECTRUM         :  OUT Adresse,Zahl
```

PAD

Befehlsart: Funktion

PAD (**Zahl**)

Beschreibung: Die PAD-Funktion gibt den Zustand eines sogenannten Touch Pad (eines angeschlossenen Zeichenbretts) wieder. Der Wert der **Zahl** liegt zwischen Ø und 7.

Beispiel:

PAD (1)

liefert den x-Wert einer berührten Position auf dem Zeichenbrett, das an den Controler A angeschlossen ist. Auf diese Weise wird die Entwicklung anspruchsvoller Grafiken unterstützt.

Kompatibilitätsliste:

	ATARI 600,800,130XE :	
	COMMODORE VC20,64 :	
	COMMODORE 16,116,PLUS4 :	
	COMMODORE 128 :	
	ENTERPRISE 64,128 :	
›	MSX-STANDARD :	PAD(Zahl)
	SCHNEIDER CPC464 :	
	SCHNEIDER CPC664,CPC6128 :	
	SINCLAIR ZX81 :	
	SINCLAIR SPECTRUM :	

PADDLE

Befehlsart: Funktion

PADDLE(**Zahl**)

Beschreibung: Die Abfrage PADDLE(**Zahl**) gibt die momentane Stellung des durch die **Zahl** bezeichneten Drehreglers wieder. Der Wert der **Zahl** muß zwischen Ø und 3 liegen.
Ergebnis ist eine Zahl zwischen 1 und 228.

Beispiel:
```
10 PRINT PADDLE (0)
20 GOTO 10
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	PADDLE(Zahl)
COMMODORE VC20,64	
COMMODORE 16,116,PLUS4	
COMMODORE 128	POT(Zahl)
ENTERPRISE 64,128	
MSX-STANDARD	PDL(Zahl)
SCHNEIDER CPC464	
SCHNEIDER CPC664,CPC6128	
SINCLAIR ZX81	
SINCLAIR SPECTRUM	

PAINT

Befehlsart: Anweisung

PAINT **Zahl 1** (**Zahl 2**,**Zahl 3**),**Zahl 4**,**Zahl 5**

Beschreibung: Mit PAINT kann der Benutzer einen geschlossenen Bereich auf dem Bildschirm mit der Farbe ausfüllen, die durch **Zahl 4** bestimmt wird. Der Cursor muß dabei in diesem Bereich liegen; seine Stellung bestimmt man mit **Zahl 2** (= Wert auf der x-Achse bzw. die Spalte) und **Zahl 3**

(= Wert auf der y-Achse bzw. die Zeile). **Zahl 5** bestimmt die Farbe der Grenzlinie um den eingefärbten Bereich. **Zahl 1** gibt die letzte Cursor-Position des Grafik-Cursors als neuen Ursprung für die Koordinaten wieder. Auf sie kann auch verzichtet werden.

Das Befehlsformat wirkt nur, wenn der Grafik-Modus mit SCREEN 3 (siehe unter SCREEN) festgelegt wurde.

Beispiel:

```
10 SCREEN 3
20 CIRCLE (140,85),60,10,,,8,8
30 PAINT (140,85),8,10
40 GOTO 40

READY.
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	PAINT Zahl3,Zahl1,Zahl2
› COMMODORE 128	:	PAINT Zahl3,Zahl1,Zahl2
ENTERPRISE 64,128	:	PLOT Zahl1,Zahl2, PAINT
› MSX-STANDARD	:	PAINT STEP Zahl1,Zahl2,Zahl3,Zahl4
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

PAPER

Befehlsart: Anweisung

PAPER **Zahl 1,Zahl 2**

Beschreibung: Mit dem Befehl PAPER (Papier) legen wir die Hintergrundfarbe fest. **Zahl 2** gibt die Nummer des Farbstiftes an, mit dem der Hintergrund ausgefüllt wird.

Vor Anwendung dieses Befehls sollte man mit dem INK-Befehl (siehe dort) die Farbstifte definieren. Falls der mit **Zahl 2** angegebene Farbstift nicht zulässig ist, wird ein anderer - verwendbarer - ausgewählt.

PAPER

Zahl 1 gibt die Kanalnummer des Ein-/Ausgabegerätes wieder. Sie wird im Normalfall nicht benötigt, so daß der Befehl das Format hat

PAPER Zahl 2

Beispiel:

```
10 INK 3,13
20 PAPER 3
30 PRINT "Die neue Hintergrundfarbe ist nun weiss."
40 PRINT
50 PRINT "Druecke die ENTER-Taste."
60 INPUT a$
70 CLS
```

Anmerkung: Wird der Befehl vom Computer ausgeführt, sieht man zunächst einmal gar nichts. Erst wenn man einen PRINT-Befehl ausführt, werden die überschriebenen Zeichen mit der neuen Hintergrundfarbe ausgefüllt. Der ganze Bildschirm ist also erst dann mit der neuen Hintergrundfarbe ausgefüllt, wenn nach dem PAPER-Befehl der ganze Bildschirm überschrieben oder wenn ein CLS-Befehl (siehe dort) ausgeführt wurde.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE        :  SETCOLOR 4,Zahl2,5
  COMMODORE V64              :  POKE 53281,Zahl2
  COMMODORE 16,116,PLUS4     :  COLOR 0,Zahl2
  COMMODORE 128              :  COLOR 0,Zahl2
  ENTERPRISE 64,128          :  SET PAPER Zahl2
  MSX-STANDARD               :  COLOR ,Zahl2
> SCHNEIDER CPC464           :  PAPER #Zahl1,Zahl2
> SCHNEIDER CPC664,CPC6128   :  PAPER #Zahl1,Zahl2
  SINCLAIR ZX81              :
> SINCLAIR SPECTRUM          :  PAPER Zahl2
```

PDL

Befehlsart: Funktion

Beschreibung: Der MSX-Befehl PDL entspricht der PADDLE-Funktion in anderen BASIC-Dialekten (z. B. ATARI).

Kompatibilitätsliste:

ATARI 600,800,130XE	:	PADDLE(Zahl)
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	POT(Zahl)
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	PDL(Zahl)
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

PEEK

Befehlsart: Funktion

PEEK(**Adresse**)

Beschreibung: Mit der PEEK-Funktion wird jene Speicherzelle nach ihrem Inhalt abgefragt, die durch die **Adresse** angegeben ist.

Beispiel:

```
10 PRINT "Minimonitor"
12 PRINT
14 PRINT "Mit Hilfe des kleinen Programmes kann"
16 PRINT "man den Speicherinhalt des Rechners"
18 PRINT "abfragen."
20 PRINT
22 INPUT "Anfangsadresse des Speicherbereiches",a
24 IF a<0 THEN PRINT "(0-65535)":GOTO 22
26 IF a>65535 THEN PRINT "(0-65535)":GOTO 22
28 a=INT(a/4)*4
30 PRINT "Adr";a;"  ";
32 FOR i =a TO a+3
34 PRINT PEEK (i);
36 NEXT i
38 a$=INKEY$:IF a$="" THEN 38
40 a=a+4
42 GOTO 30
```

PEEK

Mit diesem Programm wird der Speicherinhalt ab einer bestimmten, vom Benutzer eingegebenen Speicherzelle abgefragt und auf dem Bildschirm angezeigt. Durch Drücken der Leertaste oder einer anderen Taste werden die jeweils nächsten 4 Werte auf den Bildschirm gebracht.

Besonderheiten: Bei ATARI- und Commodore-Computern muß die Zeile 38 wie folgt lauten:

```
38 GET a$:IF a$='''' THEN 38
```

Anmerkung: Durch den Befehl POKE (siehe dort) kann man einer bestimmten Speicheradresse einen Inhalt zuordnen. Auf diese Weise ist es zum Beispiel möglich, sein BASIC-Programm in den Rechner-Speicher zu "poken", damit es LIST-geschützt ist (siehe LIST). Dazu muß man sich zunächst mit dem PEEK-Befehl eine Zahlendatei erstellen und sie abspeichern.

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  PEEK(Adresse)
> COMMODORE VC20,64         :  PEEK(Adresse)
> COMMODORE 16,116,PLUS4    :  PEEK(Adresse)
> COMMODORE 128             :  PEEK(Adresse)
> ENTERPRISE 64,128         :  PEEK(Adresse)
> MSX-STANDARD              :  PEEK(Adresse)
> SCHNEIDER CPC464          :  PEEK(Adresse)
> SCHNEIDER CPC664,CPC6128  :  PEEK(Adresse)
> SINCLAIR ZX81             :  PEEK Adresse
> SINCLAIR SPECTRUM         :  PEEK Adresse
```

PEN

Befehlsart: Funktion

PEN(**Zahl**)

Beschreibung: Es werden die Bildschirmkoordinaten des Lightpen (Lichtgriffel) abgefragt, der auf den Bildschirm gehalten wird. Dazu kann man mit der entsprechenden **Zahl** fragen, ob ein Lightpen aktiviert ist oder nicht.

Die **Zahl** gibt an, was abgefragt werden soll. Möglich sind **Zahl**en zwischen Ø und 4. Dabei bedeutet

Zahl

Ø	die x-Koordinate (Spalte) auf dem Bildschirm (6Ø - 32Ø)
1	die y-Koordinate (Zeile) auf dem Bildschirm (5Ø - 2ØØ)
2	die x-Koordinate (Spalte) auf dem 8Ø-Zeichen-Bildschirm
3	die y-Koordinate (Zeile) auf dem 8Ø-Zeichen-Bildschirm
4	die Aktivierung des Lightpen **Zahl 4** gibt eine 1 zurück, sobald der Lightpen aktiviert ist, und eine Ø, falls er nicht aktiviert ist.

Beispiel:

```
10 REM *POSITION EINES LIGHTPEN
20 DO UNTIL PEN(4):LOOP
30 PRINT "X-KOORDINATE"PEN(0),"Y-KOORDINATE"PEN(1)
40 GOTO 20

READY.
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
› COMMODORE 128             :  PEN(Zahl)
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

PEN

Befehlsart: Kommando

PEN#**Zahl 1**,**Zahl 2**

Beschreibung: Mit dem Befehl PEN (= Stift) wird die Farbe des Zeichenstiftes neu bestimmt. Dazu sollte man vorher anhand des INK-Befehls (siehe dort) die Farbstift-Farben belegen. Durch den Befehl PEN wird der Farbstift mit der Nr. **Zahl 2** angewählt.

Im Normalfall kann man die Kanalnummer **Zahl 1** weglassen, da automatisch der Bildschirm angesprochen wird. Der Befehl lautet wie folgt:

PEN **Zahl 2**

Beispiel:

```
10 INPUT a$
20 FOR i=1 TO LEN (a$)
30 PEN INT(RND*16)
40 PRINT MID$(a$,i,1);
50 NEXT i
60 PRINT
70 GOTO 10
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :  COLOR 1,Zahl2
  COMMODORE 128             :  COLOR 5,Zahl2
  ENTERPRISE 64,128         :  SET INK Zahl2
  MSX-STANDARD              :  COLOR Zahl2
> SCHNEIDER CPC464          :  PEN #Zahl1,Zahl2
> SCHNEIDER CPC664,CPC6128  :  PEN #Zahl1,Zahl2
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :  INK Zahl2
```

PI

Befehlsart: Konstante

PI

Beschreibung: PI ist die mathematische Konstante pi (π) = 3.14159265346825. Es ist die Kreiszahl, die für die Berechnung kreisförmiger Körper und für die Winkelfunktionen benötigt wird.

Beispiele:

```
? PI
 3.14159265
READY
```

```
10 PRINT "Berechnung des Kreisinhaltes."
20 PRINT "Geben Sie den Radius des zu berechnenden Kreises
                                          in Zentimeter ein: "
30 INPUT r
40 PRINT:PRINT "Der Inhalt des Kreises betraegt ";r^2*PI;"
                                          Quadratzentimeter."
```

Anmerkung zur Kompatibilität: Bei Rechnern, in deren BASIC-Wortschatz diese Konstante nicht vorkommt, definiert man sie am besten zu Beginn eines entsprechenden Programmes.

Beispiel:

```
10 P= 3.14159265
20 PRINT "BERECHNUNG DES KREISINHALTES"
30 PRINT "GEBEN SIE DEN RADIUS DES ZU BERECHNENDEN KREISES
                                              IN ZENTIMETER EIN: "
40 INPUT R
50 PRINT:PRINT "DER INHALT DES KREISES BETRAEGT ";R^2*P;"
                                                QUADRATZENTIMETER."
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  PI
  MSX-STANDARD              :
> SCHNEIDER CPC464          :  PI
> SCHNEIDER CPC664,CPC6128  :  PI
> SINCLAIR ZX81             :  PI
> SINCLAIR SPECTRUM         :  PI
```

PING

Befehlsart: Anweisung

PING

Beschreibung: Es wird ein kurzer Ton erzeugt. Der Befehl entspricht der BEEP-Anweisung (siehe dort).

Beispiel:

```
1ØØ  INPUT "ERSTE ZAHL";A
11Ø  INPUT "ZWEITE ZAHL";B
12Ø  PRINT A;"  ";B;" = "
13Ø  INPUT C
14Ø  IF C=A B THEN PRINT "RICHTIG":GOTO 1ØØ
15Ø  PING: PRINT "FALSCH":GOTO 13Ø
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  PING
  MSX-STANDARD              :  BEEP
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :  BEEP Tondauer,Tonhoehe
```

PLAY

Befehlsart: Anweisung

PLAY **String**

Beschreibung: Mit Hilfe dieser Anweisung läßt sich die Ton-Ausgabe sehr einfach steuern. Mit den **String**-Konstanten kann man bis zu 3 verschiedene Ton-Kanäle ansprechen. Dazu werden weitere Daten nach Belieben angegeben, so etwa Tempo, Oktaven, Tonlänge usw.

Beispiel:

```
10 PLAY "T8003L4CDEFG2.RAB04CDC2"
20 PLAY "A","C","E"
30 GOTO 10

READY.
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  PLAY Parameterliste
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  PLAY Option
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

PLOT

Befehlsart: Kommando

PLOT **Zahl 1,Zahl 2,Zahl 3**

Beschreibung: Mit diesem Befehl kann man einen Punkt auf dem Bildschirm plotten (zeichnen). Dabei wird mit den Zahlen **Zahl 1** und **Zahl 2** die Position des Punktes und die neue Position des Grafikcursors festgelegt.

Die Angabe der Farbstiftnummer **Zahl 3** kann man weglassen. Falls man diese angibt, so kann man mit dem INK-Befehl (siehe dort) die Farben der Stifte festlegen.

Beispiel:

```
10 INPUT "radius";r
20 CLS
30 MODE 2
40 PRINT "kreis mit dem radius"r
50 FOR i=0 TO 2*PI STEP PI/10/r
60 PLOT r+5+INT(r*COS(i)+0.5),r+5+INT(r*SIN(i)+0.5)
70 NEXT i
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	PLOT Zahl1,Zahl2
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	PLOT Zahl1,Zahl2
MSX-STANDARD	:	PSET Zahl1,Zahl2,Zahl3
› SCHNEIDER CPC464	:	PLOT Zahl1,Zahl2,Zahl3
› SCHNEIDER CPC664,CPC6128	:	PLOT Zahl1,Zahl2,Zahl3
SINCLAIR ZX81	:	
› SINCLAIR SPECTRUM	:	PLOT Zahl1,Zahl2

PLOTR

Befehlsart: Anweisung

PLOTR **Zahl 1,Zahl 2,Zahl 3**

Beschreibung: Ein Punkt wird auf dem Bildschirm gezeichnet ("geplottet"). **Zahl 1** gibt den Wert auf der x-Achse im Verhältnis zur alten Cursorstellung wieder, **Zahl 2** den Wert auf der y-Achse.

Zahl 3 bezeichnet die Farbstiftnummer für den Punkt. Sie kann weggelassen werden, wenn man keine Veränderung der Farbe wünscht. Wenn wir **Zahl 3** angeben, lassen sich anhand des INK-Befehls (siehe dort) die Farbstifte festlegen.

Beispiel:

```
10 FOR I=3 TO 20
20 KEY DEF I,1
30 NEXT I
40 MODE 2
50 PRINT "MINI MAL PROGRAMM"
60 PLOT 150,150
70 A$=INKEY$:IF A$<"1" OR A$>"9" THEN 70
80 X=0:Y=0
90 IF INSTR("789",A$)>0 THEN Y=1
100 IF INSTR("123",A$)>0 THEN Y=-1
110 IF INSTR("369",A$)>0 THEN X=1
120 IF INSTR("147",A$)>0 THEN X=-1
130 PLOTR X,Y
140 GOTO 70
```

Mit diesem kleinen Malprogramm kann der Benutzer beliebige Figuren auf dem Bildschirm malen. Zur Angabe der jeweiligen Richtungen werden die Zahlen 1 bis 9 auf dem Zehnerblock verwendet.

Die Zeilen 1Ø bis 3Ø legen die Dauerfunktion für den Zehnerblock fest.

Zeile 4Ø schaltet den hochauflösenden Bildschirm-Modus ein.

Die Zeilen 7Ø bis 14Ø bewirken die Tastenabfrage und das Festlegen des neuen Punktes auf dem Bildschirm.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	PSET STEP Zahl1,Zahl2,Zahl3
› SCHNEIDER CPC464	:	PLOTR Zahl1,Zahl2,Zahl3
› SCHNEIDER CPC664,CPC6128	:	PLOTR Zahl1,Zahl2,Zahl3
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

POINT

Befehlsart: Kommando

POINT #**Zahl 1**,**Zahl 2**,**Zahl 3**

Beschreibung: Der Schreib-/Lese-Zeiger wird auf eine Stelle innerhalb einer Datei gesetzt, die durch **Zahl 2** und **Zahl 3** gekennzeichnet ist.

Die **Zahl 1** gibt die Nummer an, unter der die Datei vorher eröffnet wurde. Sie wird mit dem OPEN-Befehl (siehe dort) geöffnet. **Zahl 1** kann Werte zwischen 1 und 7 annehmen.

Die **Zahl 2** gibt die Nummer des Diskettensektors an, auf den der Zeiger gerichtet werden soll.

Innerhalb des Datensatzes, der durch die **Zahl 2** angesprochen ist, gibt **Zahl 3** die Nummer des entsprechenden Bytes an, auf das der Zeiger gerichtet werden soll.

Wird der Datenzeiger auf einen nicht vorhandenen Datensatz gestellt, erfolgt eine Fehlermeldung.

Dieser Befehl ähnelt dem Befehl RECORD (siehe dort).

Beispiel:

```
POINT #5,1,2
```

Anmerkung zur Kompatibilität: Das Kommando POINT ist nicht mit der Funktion POINT beim Sinclair-Computer zu verwechseln (siehe dort).

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	POINT #Zahl1,Zahl2,Zahl3
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	RECORD #Zahl1,Zahl2,Zahl3
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

POINTER

Befehlsart: Funktion

POINTER(**Variablenname**)

Beschreibung: Die Funktion gibt die Adresse einer Variablen mit dem **Variablennamen** zurück. Die Adresse, die man erhält, ist das erste Byte einer Variablen. Die meisten Variablen sind Mehr-Byte-Variablen (INTEGER = 2 Byte, STRING = 1...255 Byte).

Mit dem POINTER-Befehl kann man nur die Adressen von Variablen abfragen, die bereits mit einem Wert belegt worden sind.

Beispiel:

```
10 INPUT "GEBEN SIE BITTE EINE ZAHL EIN",ZA%
20 I=POINTER(ZA%)
30 BANK 1:X=PEEK(I)*256+PEEK(I+1)
40 PRINT "ICH HABE IHRE ZAHL IM SPEICHER GEFUNDEN."
50 PRINT "SIE LAUTET:"X" UND STEHT IN DEN"
60 PRINT "SPEICHERZELLEN"I"BIS"I+1
70 PRINT "IN DER SPEICHERBANK 1"
```

Anmerkung: Die vom Computer übermittelte Adresse bezieht sich beim Commodore 128 auf die Speicherbank 1, in der alle Variablen abgespeichert sind.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  POINTER(Variable)
  ENTERPRISE 64,128         :
  MSX-STANDARD              :  VARPTR(Variable)
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

POKE

Befehlsart: Kommando

POKE **Zahl 1**,**Zahl 2**

Beschreibung: Eine Speicherzelle mit der durch **Zahl 1** gekennzeichneten Adresse wird mit dem Byte **Zahl 2** beschrieben. Die Adresse muß einen Wert im Bereich von Ø bis 65535 aufweisen. Der Wert für **Zahl 2** liegt zwischen Ø und 255.

Beispiel:

```
POKE 65535,255
```

Anmerkung zur Kompatibilität: Der POKE-Befehl ist aufgrund des geringen BASIC-Wortschatzes für den Commodore 64 besonders wichtig. Mit ihm kann man fehlende Befehle simulieren.

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  POKE Zahl1,Zahl2
> COMMODORE VC20,64         :  POKE Zahl1,Zahl2
> COMMODORE 16,116,PLUS4    :  POKE Zahl1,Zahl2
> COMMODORE 128             :  POKE Zahl1,Zahl2
> ENTERPRISE 64,128         :  POKE Zahl1,Zahl2
> MSX-STANDARD              :  POKE Zahl1,Zahl2
> SCHNEIDER CPC464          :  POKE Zahl1,Zahl2
> SCHNEIDER CPC664,CPC6128  :  POKE Zahl1,Zahl2
> SINCLAIR ZX81             :  POKE Zahl1,Zahl2
> SINCLAIR SPECTRUM         :  POKE Zahl1,Zahl2
```

POP

Befehlsart: Anweisung

POP

Beschreibung: Durch den POP-Befehl wird die im Stack gespeicherte Rücksprungadresse des zuletzt ausgeführten FOR...TO, GOSUB oder ON...GOSUB (siehe jeweils dort) gelöscht. Wichtig: Es wird jeweils nur eine Rücksprungadresse gelöscht.

Mit diesem Befehl kann man also aus einer FOR...TO-Schleife herausspringen, ohne eine Fehlermeldung zu riskieren.

Beim GOSUB- und ON GOSUB-Befehl wird das RETURN (siehe dort) zum letzten ausgeführten Befehl "vergessen". Der Rechner interpretiert nun GOSUB als GOTO (siehe dort) und ON...GOSUB als ON...GOTO (siehe dort).

Falls keiner der drei Befehle in einem "offenen Zustand" ist, erfolgt auch keine Fehlermeldung. Unter dem "offenen Zustand" versteht man, daß bei GOSUB und ON...GOSUB noch kein RETURN ausgeführt wurde und bei FOR...TO noch nicht die letzte Rückkehr, bedingt durch das NEXT (siehe dort), erreicht wurde.

Beispiel:

```
10 PRINT "GIB EINE ZAHL VON 1 BIS 50 EIN."
20 INPUT ZA
30 FOR I=0 TO 50
40 PRINT I
50 IF I=ZA THEN POP:GOTO 80
60 NEXT I
70 GOTO 30
80 PRINT "ICH HABE BIS ZU DEINER ZAHL GEZAEHLT."
90 END
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	: POP
COMMODORE VC20,64	:
COMMODORE 16,116,PLUS4	:
COMMODORE 128	:
ENTERPRISE 64,128	:
MSX-STANDARD	:
SCHNEIDER CPC464	:
SCHNEIDER CPC664,CPC6128	:
SINCLAIR ZX81	:
SINCLAIR SPECTRUM	:

POS

Befehlsart: Funktion

POS (**Kanalnummer**)

Beschreibung: Die Funktion gibt die Spaltenposition des Textcursors an (Wert auf der x-Achse)..

Die Kanalnummer muß dabei auf jeden Fall angegeben werden. Der Hauptbildschirm hat die **Kanalnummer#Ø.**

Wenn man die Position des Drucker-Schreibkopfes überprüfen will, muß die **Kanalnummer**#8 angegeben werden.

Beispiele:

```
? "schneider buch";POS (#Ø)
 15
READY
```

```
10 CLS
20 FOR y= 1 TO 208
30 PRINT "*";
40 IF POS(#0)= 4 THEN PRINT "---";
50 IF POS(#0) = 12 THEN PRINT "###";
60 IF POS(#0)= 20 THEN PRINT "====";
70 IF POS(#0)=27 THEN PRINT
80 NEXT y
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
> COMMODORE VC20,64	:	POS(0)
> COMMODORE 16,116,PLUS4	:	POS(0)
> COMMODORE 128	:	POS(0)
ENTERPRISE 64,128	:	
> MSX-STANDARD	:	POS(0)
> SCHNEIDER CPC464	:	POS(#Kanalnummer)
> SCHNEIDER CPC664,CPC6128	:	POS(#Kanalnummer)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

POSITION

Befehlsart: Anweisung

POSITION **Zahl 1,Zahl 2**

Beschreibung: Der Textcursor wird auf die neue, von **Zahl 1** und **Zahl 2** bestimmte Stelle gesetzt.

Zahl 1 entspricht dabei der x-Achse (Spalte bzw. Abszisse), und **Zahl 2** der y-Achse (Zeile bzw. Ordinate).

Diese neue Cursorposition wird benutzt, sobald die nächste Ausgabe auf dem Bildschirm erscheint.

Beispiel:

```
10 FOR N=1 TO 17
20 POSITION N*2,N
30 PRINT "ATARI"
40 NEXT N
```

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  POSITION Zahl1,Zahl2
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :  LOCATE Zahl1,Zahl2
  COMMODORE 128             :  LOCATE Zahl1,Zahl2
  ENTERPRISE 64,128         :
  MSX-STANDARD              :  LOCATE Zahl1,Zahl2,Zahl
  SCHNEIDER CPC464          :  LOCATE #Zahl,Zahl1,Zahl2
  SCHNEIDER CPC664,CPC6128  :  LOCATE #Zahl,Zahl1,Zahl2
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

POT

Befehlsart: Funktion

POT(**Zahl**)

Beschreibung: Die Funktion fragt die Drehregler ("Paddles") ab, die am Computer angeschlossen sind.

Die **Zahl** gibt dabei an, welches der vier Paddles, die mit dem Rechnern verbunden werden können, durch diesen Befehl angesprochen wird.

Der Wert, den man erhält, liegt zwischen Ø und 255. Falls der Feuerknopf des Paddles gedrückt wurde, ist der Wert um 256 erhöht.

Beispiele:

```
I=POT (1)

PRINT POT (2)
```

Anmerkungen zur Kompatibilität: Beim ATARI lautet der entsprechende Befehl PADDLE (siehe dort).

Beim Commodore 64 muß die Funktion mit dem PEEK-Befehl (siehe dort) ersetzt werden:

```
PADDLE 1/2  PORT 1  PEEK(54297)                PEEK(54298)
PADDLE 3/4  PORT 2  PEEK(5632Ø):PEEK(54297)    PEEK(5632Ø):PEEK(54298)
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	PADDLE(Zahl)
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
› COMMODORE 128	:	POT(Zahl)
ENTERPRISE 64,128	:	
MSX-STANDARD	:	PDL(Zahl)
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

PRESET

Befehlsart: Anweisung

PRESET STEP (**Zahl 1**,**Zahl 2**),**Zahl 3**

Beschreibung: Anhand der PRESET-Anweisung wird der durch **Zahl 1** (= x-Richtung) und **Zahl 2** (= y-Richtung) gekennzeichnete Bildschirmpunkt mit der Hintergrundfarbe gelöscht oder mit einer durch **Zahl 3** angegebenen Farbe eingefärbt. Der Wert für **Zahl 3** kann zwischen Ø und 15 liegen.

Ist der Zusatz STEP angegeben (siehe auch unter CIRCLE), werden die Koordinaten für den Punkt in Abhängigkeit von der letzten Stellung des Grafik-Cursors errechnet.

Beispiel:

```
1ØØ FOR X=1 TO 1Ø
11Ø PSET (X,1Ø)
12Ø PRESET (X,1Ø)
13Ø NEXT X
14Ø GOTO 1ØØ
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  PRESET STEP (Zahl1,Zahl2),Zahl3
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

PRINT

Befehlsart: Anweisung

PRINT
PRINT **Rechenausdruck**
PRINT "**Textausdruck**"
PRINT **Variablenname**

Beschreibung: Der Befehl bewirkt die Ausgabe des **Textausdrucks** oder des Inhaltes der Variable **Variablenname** auf dem Bildschirm bzw. die Auflösung des **Rechenausdrucks**. Hat man eine PRINT-Anweisung ohne Parameter angegeben, so wird eine Leerzeile ausgegeben.

Beispiel:

```
10 PRINT "Wir ueben das Einmaleins:"
20 PRINT "Welche Zahl wollen wir ueben";:INPUT a
30 FOR b=1 TO 20
40 PRINT a;" * ";b;" = ";:INPUT c
50 IF c=a*b THEN PRINT "richtig":GOTO 80
60 PRINT "leider falsch"
70 PRINT a;" * ";b;" = ";a*b:GOTO 80
80 PRINT:PRINT:PRINT:NEXT b
90 PRINT "Weiter (J/N)?"
100 INPUT q$:IF q$="J"OR Q$="j" THEN GOTO 10
110 END
```

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  PRINT Ausgabeliste
› COMMODORE VC20,64         :  PRINT Ausgabeliste
› COMMODORE 16,116,PLUS4    :  PRINT Ausgabeliste
› COMMODORE 128             :  PRINT Ausgabeliste
› ENTERPRISE 64,128         :  PRINT Ausgabeliste
› MSX-STANDARD              :  PRINT Ausgabeliste
› SCHNEIDER CPC464          :  PRINT Ausgabeliste
› SCHNEIDER CPC664,CPC6128  :  PRINT Ausgabeliste
› SINCLAIR ZX81             :  PRINT Ausgabeliste
› SINCLAIR SPECTRUM         :  PRINT Ausgabeliste
```

PRINT#

Befehlsart: Anweisung

PRINT# **Zahl,"Textausdruck"**
PRINT# **Zahl,Variablenname**

Beschreibung: Der Befehl gibt Daten in einem vorher mit OPEN (siehe dort) eröffneten Kanal aus. Die Datei bzw. das Peripheriegerät wird durch die **Zahl** gekennzeichnet.

Beispiel:

```
1Ø A$="Computer"
2Ø B$="Buecher"
3Ø PRINT#1,"SCHNEIDER";A$;B$
```

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  PRINT #Kanal,Ausgabeliste
› COMMODORE VC20,64         :  PRINT #Kanal,Ausgabeliste
› COMMODORE 16,116,PLUS4    :  PRINT #Kanal,Ausgabeliste
› COMMODORE 128             :  PRINT #Kanal,Ausgabeliste
› ENTERPRISE 64,128         :  PRINT #Kanal,Ausgabeliste
› MSX-STANDARD              :  PRINT #Kanal,Ausgabeliste
› SCHNEIDER CPC464          :  PRINT #Kanal,Ausgabeliste
› SCHNEIDER CPC664,CPC6128  :  PRINT #Kanal,Ausgabeliste
  SINCLAIR ZX81             :
› SINCLAIR SPECTRUM         :  PRINT #Kanal,Ausgabeliste
```

PRINT USING

Befehlsart: Anweisung

PRINT USING **String,Ausgabeliste**

Beschreibung: Der Befehl ermöglicht die Ausgabe von formatierten Zahlen und Texten. Mit **String** wird eine Formatschablone entworfen, die von Computertyp zu Computertyp verschiedene Elemente verwendet. Der Befehl ist komplex genug, um ihn anhand des Handbuches in allen seinen Möglichkeiten auszuprobieren.
Bedeutung der **Ausgabeliste** siehe PRINT.

Beispiel:

```
10 A$="######.##"
20 B$="!!"
30 PRINT USING A$;56270.0931;
40 PRINT USING B$;"meter"
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  PRINT #Kanal USING String;Ausgabeliste
> COMMODORE 128             :  PRINT #Kanal USING String;Ausgabeliste
> ENTERPRISE 64,128         :  PRINT #Kanal,USING String:Ausgabeliste
> MSX-STANDARD              :  PRINT USING String;Ausgabeliste
> SCHNEIDER CPC464          :  PRINT #Kanal;USING String;Ausgabeliste
> SCHNEIDER CPC664,CPC6128  :  PRINT #Kanal;USING String;Ausgabeliste
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

PROGRAM

Befehlsart: Kommando

PROGRAM **Programmname(Variablenliste)**

Beschreibung: Durch dieses Kommando wird ein **Programmname** für ein Programm festgelegt. Dieses Programm kann man mit

CHAIN (siehe dort) von einem anderen Programm aus aufrufen.

Anhand der **Variablenliste** lassen sich Werte von Variablen aus dem vorhergehenden Programm übernehmen. Sie kann bei Nichtbedarf auch weggelassen werden.

Beispiel:

```
1Ø   PROGRAM "SUPER"
2Ø   PRINT "DIES IST DAS PROGRAMM SUPER."
3Ø   CHAIN "HYPER"

EDIT 1

1Ø   PROGRAM "HYPER"
2Ø   PRINT "NUN SIND WIR IM PROGRAMM HYPER."
3Ø   CHAIN "SUPER"

RUN

NUN SIND WIR IM PROGRAMM HYPER.
DIES IST DAS PROGRAMM SUPER.
NUN SIND WIR IM PROGRAMM HYPER.
DIES IST DAS PROGRAMM SUPER.
 .
 .
 .
```

Anmerkung: Das Kommando PROGRAM sollte in der ersten Zeile des Programmes stehen.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	PROGRAM Name(Variablenliste)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

PSET

Befehlsart: Anweisung

PSET **Zahl 1,Zahl 2,Farbstiftnummer**

Beschreibung: PSET setzt einen Punkt auf dem Bildschirm mit den Koordinaten **Zahl 1** (x-Achse bzw. Abszisse) und **Zahl 2** (y-Achse bzw. Ordinate) sowie der **Farbstiftnummer.**

Beispiel:

```
11Ø PSET 45,75
```

Kompatibilitätsliste: Siehe PSET STEP

PSET STEP

Befehlsart: Anweisung

PSET STEP **(Zahl 1,Zahl 2),Zahl 3**

Beschreibung: Mit diesem Befehl läßt sich ein Punkt auf eine Bildschirmstelle bringen, die durch **Zahl 1** (= x-Achse) und **Zahl 2** (= y-Achse) bestimmt wird.

Zahl 3 bezeichnet die Farbnummer. Ohne Eingabe von **Zahl 3** nimmt der Rechner automatisch die augenblickliche Vordergrundfarbe an.

Den Zusatz STEP kann man wie beim CIRCLE-Befehl (siehe dort) handhaben. Damit werden die Koordinaten für den Punkt in Abhängigkeit von der letzten Stellung des Grafik-Cursors errechnet.

Beispiel:

```
10 SCREEN 2
20 FOR X= 0 TO 255
30 PSET(X+1,100):REM zeichnet einen Punkt
40 PRESET(X,100):REM loescht den zuvor gezeichneten punkt
50 NEXT X
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  PSET STEP (Zahl1,Zahl2),Zahl3
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

PTRIG

Befehlsart: Funktion

PTRIG(**Zahl**)

Beschreibung: Die Abfrage PTRIG gibt Auskunft darüber, ob der Taster ("Feuerknopf") des durch die **Zahl** bezeichneten Drehreglers (Paddle) gedrückt ist. Die **Zahl** kann dabei Werte zwischen Ø und 3 annehmen.

Liefert die Abfrage eine Ø, so wurde der Taster betätigt.

Beispiel:

```
100 IF PTRIG (0)=0 THEN PRINT "TASTE GEDRUECKT."
110 GOTO 100
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	PTRIG(Zahl)
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

PUDEF

Befehlsart: Anweisung

PUDEF **String**

Beschreibung: Mit dieser Anweisung kann man bis zu vier Steuerzeichen des Befehls PRINT USING (siehe dort) abändern.

Der **String** besteht aus maximal 4 Zeichen, die für die Änderung des PRINT USING-Formats verantwortlich sind.

Die vier Steuerzeichen sind normalerweise

1. Füllzeichen (Leerzeichen)
2. Komma
3. Dezimalpunkt
4. Dollarzeichen

Beispiel:

```
220 PUDEF ;$
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	PUDEF String
› COMMODORE 128	:	PUDEF String
ENTERPRISE 64,128	:	IMAGE:String
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

PUT

Befehlsart: Anweisung

PUT **Zahl 1,Zahl 2**

Beschreibung: Mit dem PUT-Befehl kann der Anwender ein Zeichen in einer geöffneten Datei ablegen. Er ist der Umkehrbefehl zu GET (siehe dort).

Die **Zahl 1** gibt die Kanalnummer der Datei an, die durch den OPEN-Befehl (siehe dort) geöffnet wurde. Sie kann einen Wert zwischen 1 und 7 annehmen.

Die **Zahl 2** gibt den Wert des Zeichens an, das in eine Datei geschrieben werden soll. Der Wert von **Zahl 2** muß zwischen Ø und 255 liegen. (Andernfalls sorgt der Rechner selbst dafür, daß sie in den Bereich gelangt.)

Mit Hilfe der Funktion ASC (siehe dort) kann man auch Textzeichen in der Datei ablegen, indem man sie zuvor anhand dieser Funktion in numerische Werte umwandelt.

Beispiel:

```
PUT 1,65
```

Es wird die Zahl 65 über Kanal 1 eingegeben.

```
13Ø PUT ASC ("S")
14Ø PUT ASC ("C")
15Ø PUT ASC ("B")
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	PUT #Zahl1,Zahl2
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

PUT SPRITE

Befehlsart: Anweisung

PUT SPRITE **Zahl 1** STEP (**Zahl 2**,**Zahl 3**),**Zahl 5**,**Zahl 6**

Beschreibung: Mit dem PUT SPRITE-Befehl wird ein bestimmtes Sprite-Muster an der durch **Zahl 2** (= x-Richtung) und **Zahl 3** (= y-Richtung) gekennzeichneten Stelle angezeigt. **Zahl 1** gibt die Sprite-Ebene an. Der Wert kann zwischen Ø und 31 liegen.

Zahl 5 wählt die Farbe aus. Mit **Zahl 6** schließlich wird die Sprite-Nummer angegeben. Sie kann bei einem hochauflösenden Bildschirm-Modus zwischen Ø und 63, beim niederauflösenden Bildschirm-Modus zwischen Ø und 255 liegen.

Der Zuatz STEP wird wie beim CIRCLE-Befehl (siehe dort) gehandhabt.

Beispiel:

```
10 SCREEN 2: REM Fliegendes Gesicht
20 SPRITE $(1)=CHR$(&C3)+CHR$(&DB)+CHR$(&18)+CHR$(&18)+CHR$(&DB)+CHR$(&C3)+CHR$(
&7E)
30 X=0:Y=0:DX=1:DY=1
40 PUT SPRITE 0,(X,Y),,1
50 X=X+DX:Y=Y+DY
60 IF X>250 OR X<0 THEN DX=-DX
70 IF Y>190 OR Y<0 THEN DY=-DY
80 GOTO 40
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :  MOVSPR Zahl5,Zahl2,Zahl3
                               MOVSPR Zahl5,+/-Zahl2,+/-Zahl3
                               MOVSPR Zahl5,Zahl2#Zahl3
  ENTERPRISE 64,128         :
> MSX-STANDARD              :
          PUT SPRITE Zahl1(Zahl2,Zahl3)Zahl4,Zahl5
          PUT SPRITE Zahl1 STEP(Zahl2,Zahl3)Zahl4,Zahl5
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

RAD

Befehlsart: Funktion

RAD

Beschreibung: Mit diesem Befehl wird der Rechner auf den Bogenmaß-Modus zurückgeschaltet.

Normalerweise befindet sich der Computer bereits beim Einschalten im Bogenmaß-Modus; dieser Modus ist nur für die Winkelfunktionen wichtig (siehe ACOS,ASC,ASN,ATN,COS, TAN,SIN). Verwenden muß man den Befehl RAD für den Fall, daß man zuvor mit dem Befehl DEG (siehe dort) vom Bogenmaß- auf den Winkelmaß-Modus umgeschaltet hat.

Beispiel:

```
10 DEG
20 PRINT ATN(1)
30 RAD
40 PRINT ATN(1)
```

Anmerkung: Bei Rechnern, die über diese Funktion nicht verfügen, kann vom Winkel- ins Bogenmaß umgerechnet werden, indem man die Grad-Zahl mit Ø.Ø174532 multipliziert.

Umgekehrt wird vom Bogen- ins Winkelmaß durch Multiplikation mit 57.29578 umgerechnet.

Kompatibilitätsliste:

›	ATARI 600,800,130XE	: RAD
	COMMODORE VC20,64	:
	COMMODORE 16,116,PLUS4	:
	COMMODORE 128	:
	ENTERPRISE 64,128	: OPTION ANGLE RADIANS
	MSX-STANDARD	:
›	SCHNEIDER CPC464	: RAD
›	SCHNEIDER CPC664,CPC6128	: RAD
	SINCLAIR ZX81	:
	SINCLAIR SPECTRUM	:

RAD

Befehlsart: Funktion

RAD(**Zahl**)

Beschreibung: Die Funktion wandelt die **Zahl** vom Winkelgradmaß ins Bogenmaß um.

Anmerkung: Die Funktion RAD ist nicht mit der **Anweisung** RAD zu verwechseln, die eine Umschaltung vom Winkelgradmaß in das Bogenmaß vornimmt (siehe dort).

Anmerkung zur Kompatibilität: Bei Rechnern, deren BASIC-Dialekt nicht über diesen Befehl verfügt, kann er folgendermaßen simuliert werden:

RAD(**Zahl**)

entspricht

Zahl*PI/18Ø

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
> ENTERPRISE 64,128	:	RAD(Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

RANDOMIZE

Befehlsart: Anweisung

RANDOMIZE **Zahl**

Beschreibung: Dieser Befehl setzt den Anfangswert für den Zufallsgenerator. Da die Zufallszahlen in Abhängigkeit von einem gegebenen Startwert erzeugt werden, während die Zahlenfolge immer dieselbe ist, kann mit dem RANDOMIZE-Kommando eine nahezu unwiederholbare Zahlenfolge erzeugt werden.

Beispiel:

```
10 FOR x=1 TO 100
20 RANDOMIZE x
30 PRINT RND (6)
40 NEXT x
```

Anmerkung zur Kompatibilität: Der entsprechende Befehl beim Sinclair ZX 81 hat dasselbe Format (siehe auch RND).

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  RANDOMIZE
  MSX-STANDARD              :
> SCHNEIDER CPC464          :  RANDOMIZE Zahl
> SCHNEIDER CPC664,CPC6128  :  RANDOMIZE Zahl
> SINCLAIR ZX81             :  RAND Zahl
> SINCLAIR SPECTRUM         :  RAND Zahl
```

RCLR

Befehlsart: Funktion

RCLR(**Zahl**)

Beschreibung: Mit dieser Funktion wird die Nummer der Farbe zurückgegeben, die einem bestimmten Betriebsmodus zugewiesen wurde.

Die **Zahl** gibt dabei die Nummer des Modus an. Es sind Werte von Ø bis 6 möglich.

Zahl

Ø	Hintergrund 4Ø-Zeichen-Anzeige
1	grafischer Vordergrund
2	Mehrfarben-Modus 1
3	Mehrfarben-Modus 2
4	Rand der 4Ø-Zeichen-Anzeige
5	Text der 8Ø-Zeichen-Anzeige
6	Hintergrund der 8Ø-Zeichen-Anzeige

Der Zahlenwert, den man erhält, liegt zwischen 1 und 16. Dabei entspricht

1 Schwarz	2 Weiß	3 Rot	4 Grün
5 Violett	6 Dunkelgrün	7 Blau	8 Gelb
9 Hellbraun	1Ø Braun	11 Rosa	12 Dunkelgrau
13 Grau	14 Hellgrün	15 Hellblau	16 Hellgrau

Beispiele:

```
I=RCLR(1)

PRINT RCLR(4)
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	RCLR(Zahl)
› COMMODORE 128	:	RCLR(Zahl)
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

RDOT

Befehlsart: Funktion

RDOT(**Zahl**)

Beschreibung: Auf dem Bildschirm wird die augenblickliche Stellung des Grafikcursors zurückgegeben bzw. die Farbnummer dieser Cursorposition.

Die **Zahl** kann dabei ganzzahlige Werte zwischen Ø und 2 annehmen.

Zahl

Ø	gibt die x-Achse (Abszisse bzw. Spalte) des Grafikcursors wieder.
1	gibt die y-Achse (Ordinate bzw. Zeile) des Grafikcursors wieder.
2	gibt die Farbnummer der Cursorposition wieder.

Beispiel:

```
1 REM *ZEICHNET EINEN KREIS MIT RADIUSLINIE*
10 GRAPHIC 1,1
20 CIRCLE 1,160,100,80,80
30 DRAW TO RDOT(0),100
```

Kompatibilitätsliste:

	ATARI 600,800,130XE	:
	COMMODORE VC20,64	:
›	COMMODORE 16,116,PLUS4	: RDOT(Zahl)
›	COMMODORE 128	: RDOT(Zahl)
	ENTERPRISE 64,128	:
	MSX-STANDARD	:
	SCHNEIDER CPC464	:
	SCHNEIDER CPC664,CPC6128	:
	SINCLAIR ZX81	:
	SINCLAIR SPECTRUM	:

READ

Befehlsart: Anweisung

READ **Zahlenvariable 1...ZahlenvariableX**
READ **Stringvariable 1...StringvariableX**
READ **Zahlenvariable 1,Stringvariable 1...**

Beschreibung: Die Anweisung liest die Daten im internen Datenfeld, das durch die DATA-Anweisung (siehe dort) geschaffen wurde. Das Einlesen der Daten geschieht der Reihe nach. Die READ-Anweisung liest **Zahlenvariable**n und **Stringvariable**n, die im Format des Befehls durch Kommata getrennt sind.

Beispiele:

```
10 FOR I=1TO10
20 READ A
30 PRINT A
40 NEXT I
50 DATA 1,2,3,4,5
60 DATA 6,7,8,9,10
RUN
 1
 2
 3
 4
 5
 6
 7
 8
 9
 10
READY.
```

```
10 FOR N=1 TO 10
20 READ A,A$
30 PRINT A,A$:NEXT N
40 DATA 1,EINS,2,ZWEI,3,DREI,4,VIER,5,FU
ENF
50 DATA 6,SECHS,7,SIEBEN,8,ACHT,9,NEUN,1
0,ZEHN
RUN
 1          EINS
 2          ZWEI
 3          DREI
 4          VIER
 5          FUENF
 6          SECHS
 7          SIEBEN
 8          ACHT
 9          NEUN
 10         ZEHN
READY.
```

Anmerkung: Eine einzelne READ-Anweisung kann mehrere DATA-Anweisungen bearbeiten. Mehrere READ-Anweisungen können sich auf gleiche DATA-Anweisungen beziehen.

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  READ Variablenliste
› COMMODORE VC20,64         :  READ Variablenliste
› COMMODORE 16,116,PLUS4    :  READ Variablenliste
› COMMODORE 128             :  READ Variablenliste
› ENTERPRISE 64,128         :  READ Variablenliste
› MSX-STANDARD              :  READ Variablenliste
› SCHNEIDER CPC464          :  READ Variablenliste
› SCHNEIDER CPC664,CPC6128  :  READ Variablenliste
› SINCLAIR ZX81             :  READ Variablenliste
› SINCLAIR SPECTRUM         :  READ Variablenliste
```

RECORD

Befehlsart: Kommando

RECORD **Zahl 1,Zahl 2,Zahl 3**

Beschreibung: Dieses Kommando stellt den Schreib-/Lese-Zeiger auf eine Stelle innerhalb einer relativen Datei. Diese Stelle wird durch **Zahl 2** und **Zahl 3** gekennzeichnet.

Die **Zahl 1** gibt die Nummer an, unter der die relative Datei mit dem OPEN-Befehl (siehe dort) eröffnet wurde. Sie kann Werte zwischen 1 und 255 annehmen.

Die **Zahl 2** gibt die Nummer des Datensatzes an, auf den der Zeiger gerichtet werden soll. Sie darf Werte zwischen 1 und 65535 annehmen.

Innerhalb des Datensatzes, der durch die **Zahl 2** angesprochen ist, gibt die **Zahl 3** die Nummer des Bytes an, auf das der Zeiger gerichtet werden soll. Auf diese **Zahl 3** kann man auch verzichten. Dann wird der Zeiger automatisch auf das erste Byte des durch **Zahl 2** angesprochenen Datensatzes gerichtet.

Die **Zahl 3** kann Werte zwischen 1 und 254 annehmen.

Wird der Datenzeiger auf einen nicht vorhandenen Datensatz gestellt, erhalten wir die Fehlermeldung

RECORD NOT PRESENT

Beispiele:

RECORD #1,12

RECORD #128,1,12

Kompatibilitätsliste:

ATARI 600,800,130XE	:	POINT #Zahl1,Zahl2,Zahl3
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
› COMMODORE 128	:	RECORD #Zahl1,Zahl2,Zahl3
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

REDIRECT

Befehlsart: Anweisung

REDIRECT FROM #Zahl 1 TO #Zahl 2

Beschreibung: Durch die Anweisung ist es möglich, Daten von einem Kanal, der durch **Zahl 1** bezeichnet ist, auf den durch **Zahl 2** bezeichneten Kanal umzuleiten. Gibt man für **Zahl** 2 den Wert 255 an, werden die Daten wieder vom Rechner verarbeitet.

Beispiel:

REDIRECT FROM #12Ø TO #121

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  REDIRECT FROM #Zahl1 TO #Zahl2
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

RELEASE

Befehlsart: Anweisung

RELEASE **Zahl**

Beschreibung: Der Wartezustand der einzelnen Tonkanäle läßt sich nach Bedarf aufheben. Mit dem Befehl SOUND (siehe dort) kann man die drei Tonkanäle 1, 2, 3 in einen Wartezustand versetzen, der mit RELEASE beendet wird. Die **Zahl** gibt an, welche der Tonkanäle den Wartezustand aufheben. Erlaubt sind für die **Zahl** Werte von 1 bis 7. Dabei bedeutet

Zahl

1 Freigabe von Kanal 1
2 Freigabe von Kanal 2
3 Freigabe von Kanal 1 und 2
4 Freigabe von Kanal 3
5 Freigabe von Kanal 1 und 3
6 Freigabe von Kanal 2 und 3
7 Freigabe von Kanal 1, 2 und 3

Beispiel:

```
10 SOUND 65,1000,100
20 FOR T=1 TO 5000:NEXT T
30 RELEASE 1
```

Anmerkung: Die Anwendung des Befehls setzt einige Kenntnisse im Programmieren von Musik voraus.

Anmerkung zur Kompatibilität: Bei wenigen, in diesem Buch nicht aufgenommenen BASIC-Dialekten bewirkt der Befehl RELEASE die Aufhebung des Befehls GRAB, der die Reservierung für hochauflösende Grafik freigibt.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :  RELEASE Zahl
> SCHNEIDER CPC664,CPC6128  :  RELEASE Zahl
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

REM

Befehlsart: Anweisung

REM **Textausdruck**

Beschreibung: Kommentare, die auf den eigentlichen Programmablauf keinen Einfluß haben sollen, fügen wir in das Programm-Listing mit Hilfe von Zeilen ein, vor denen der Ausdruck REM steht. In den REM-Zeilen kann man wichtige Erläuterungen unterbringen, die Aufschluß über den Programmaufbau und dessen Ablauf geben. Dadurch helfen sie bei der Erstellung und bei der Analyse eines Programmes.

Der Befehl definiert alles, was in der Zeile nach dem REM kommt, als Kommentar, also auch den Doppelpunkt. Daher läßt sich nach einer REM-Anweisung in einer Zeile keine weitere Anweisung mehr unterbringen.

Beispiel:

```
10 REM * EINMALEINS MIT 5 *
20 FOR A=1 TO 10
30 PRINT "5 * ";A;" = ";5*A
40 NEXT A
```

Es ist möglich, mehrere REM-Zeilen zu einem grafischen Bild zusammenzufügen.

Beispiel:

```
2070 REM ************************
2080 REM **                    **
2090 REM **      BLOCK 6       **
2100 REM **                    **
2110 REM ************************
2120 DATA 122, 145, 133, 189, 123, 12
```

Die REM-Anweisung darf auch am Ende einer Zeile stehen.

Beispiel:

```
10 CLS:REM BILDSCHIRM LOESCHEN
20 A$="SCHNEIDER BUCH":REM STRING DEFINIEREN
30 B$=RIGHT$(A$,4):REM TEILSTRING DEFINIEREN
40 PRINT B$:REM TEILSTRING DRUCKEN
```

Anmerkung zur Kompatibilität: In einigen wenigen BASIC-Dialekten (z. B. TI99/4A) ist das Einfügen der REM-Anweisung an das Ende einer Zeile nicht möglich.

Kompatibilitätsliste:

```
ATARI 600,800,130XE       :  REM Text
COMMODORE VC20,64         :  REM Text
COMMODORE 16,116,PLUS4    :  REM Text
COMMODORE 128             :  REM Text
ENTERPRISE 64,128         :  REM Text
MSX-STANDARD              :  REM Text
SCHNEIDER CPC464          :  REM Text
SCHNEIDER CPC664,CPC6128  :  REM Text
SINCLAIR ZX81             :  REM Text
SINCLAIR SPECTRUM         :  REM Text
```

REM

Befehlsart: Funktion

REM(**Zahl 1**,**Zahl 2**)

Beschreibung: Der Rest einer Division von **Zahl 1** durch **Zahl 2** wird zurückgegeben. Diese Funktion wirkt nur einwandfrei, wenn eine der Zahlen negativ ist. Im anderen Fall muß man auf die Funktion MOD (siehe dort) zurückgreifen.

Beispiel:

```
PRINT REM(-1,3)
-1
READY
```

Anmerkungen zur Kompatibilität: Bei den meisten Rechnern genügt die Funktion MOD, die stets das richtige Ergebnis zurückgibt. Eine Ausnahme bildet der ENTERPRISE, der den zweiten Befehl benötigt.

Die Funktion REM sollte man möglichst nicht mit der Anweisung REM (siehe dort) verwechseln, die Anmerkungen des Programmierers im Listing ermöglicht.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	REM(Zahl1,Zahl2)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

REMAIN

Befehlsart: Anweisung

REMAIN(**Zahl**)

Beschreibung: Durch diese Anweisung läßt sich der Zeitgeber abschalten, den man durch den EVERY- oder AFTER-Befehl (siehe jeweils dort) eingestellt hat. Die in Klammern angegebene **Zahl** zeigt an, welcher Zeitgeber abgeschaltet werden soll. Dabei genügen Befehle wie z=REMAIN(**Zahl**) oder PRINT REMAIN(**Zahl**), um den Zeitgeber abzustellen. PRINT REMAIN(**Zahl**) bewirkt die Ausgabe der restlichen Zeit bis zur nächsten Unterbrechung, die aber nicht mehr erfolgt, da der Zeitgeber gleichzeitig abgeschaltet wird.

Bei nicht aktivem Zeitgeber ist der Wert Ø.

Beispiel:

```
1Ø   EVERY 5Ø,1 GOSUB 5Ø
2Ø   EVERY 8Ø,2 GOSUB 1ØØ
3Ø   PRINT "aaaa";:GOTO 3Ø
4Ø   END
5Ø   PRINT "Q"
6Ø   RETURN
1ØØ  PRINT REMAIN(1)
11Ø  RETURN
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	REMAIN(Zahl)
› SCHNEIDER CPC664,CPC6128	:	REMAIN(Zahl)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

RENAME

Befehlsart: Kommando

RENAME **Programmname 1** TO **Programmname 2**,DZahl 1,UZahl 2

Beschreibung: Mit diesem Befehl wird eine Datei mit **Programmname 1** in **Programmname 2** umbenannt. Die beiden Programmnamen sind String-Größen. Gibt man sie als Text an, müssen Anführungszeichen benutzt werden. Andernfalls setzt man definierte Strings ein, die in Klammern gesetzt werden.

Bei Verwendung nur eines Laufwerkes genügt die Kurzform des Befehls im Format

RENAME**Programmname 1** TO **Programmname 2**

Zahl 1 gibt die Nummer des Laufwerkes an. Sie kann Ø oder 1 betragen.

Zahl 2 gibt die Geräteadresse des Laufwerkes an. Sie ist nur sinnvoll, wenn man zwei einzelne Diskettenlaufwerke verwendet. Der Wert von **Zahl 2** liegt normalerweise zwischen 7 und 1Ø; er darf aber auch zwischen 4 und 15 liegen.

Beispiele:

```
RENAME(a$)TO(b$)
RENAME"test" TO "versuch"
RENAME"apfel"TO"birne",DO
RENAME"birne" TO "apfel",DO,U8
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :
          RENAME "Programmname1" TO "Programmname2",DZahl1,UZahl2
> COMMODORE 128             :
          RENAME "Programmname1" TO "Programmname2",DZahl1,UZahl2
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

RENUM

Befehlsart: Kommando

RENUM **Zahl 1,Zahl 2,Zahl 3**

Beschreibung: Mit diesem Kommando versieht man ein Programm, das sich im Speicher des Rechners befindet, mit neuen Zeilennummern.

Zahl 1 gibt die Startnummer für das neue Programm an.

Zahl 2 gibt die ursprüngliche Zeilennummer an, ab der das Kommando die Umnumerierung vornimmt.

Zahl 3 gibt die Schrittweite an, in der das Programm neu numeriert wird. Der Standardwert ist hier 1Ø.

RENUM 1ØØ,1Ø,1Ø beginnt also bei der alten Zeile 1Ø mit der Umnumerierung des Programms; die erste Zeile des neuen Programms erhält die Nummer 1ØØ, und das Programm wird in Zehnerschritten numeriert.

Hat man eine der **Zahl**en nicht angegeben, wird die 1Ø als vorgegebener Schritt angenommen.

RENUM

hat folglich dieselbe Funktion wie

RENUM 1Ø,11Ø,1Ø

Bei der Neunumerierung werden in den meisten BASIC-Dialekten auch alle GOTO- und GOSUB-Anweisungen (siehe dort) aktualisiert.

Beispiel:

Aus

```
12 PRINT "      1"
14 PRINT "       2"
17 PRINT "        3"
24 PRINT "         4"
26 PRINT "          5"
34 PRINT "           6"
35 PRINT "            7"
36 PRINT
38 PRINT "EINE KLEINE HEX'"
45 PRINT "KOCHT RUEBEN.
55 FOR T=1 TO 550:NEXT:CLS
56 INPUT "NOCHMAL? >J<";A$
67 IF A$="J" THEN GOTO 12
```

wird durch
RENUM 10,12,10

```
10 PRINT "      1"
20 PRINT "       2"
30 PRINT "        3"
40 PRINT "         4"
50 PRINT "          5"
60 PRINT "           6"
70 PRINT "            7"
80 PRINT
90 PRINT "EINE KLEINE HEX'"
100 PRINT "KOCHT RUEBEN.
110 FOR T=1 TO 550:NEXT:CLS
120 INPUT "NOCHMAL? >J<";A$
130 IF A$="J" THEN GOTO 10
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	RENUMBER Zahl1,Zahl3,Zahl2
COMMODORE 128	:	RENUMBER Zahl1,Zahl3,Zahl2
ENTERPRISE 64,128	:	RENUMBER Zahl2 TO LAST AT Zahl1 STEP Zahl3
› MSX-STANDARD	:	RENUM Zahl1,Zahl2,Zahl3
› SCHNEIDER CPC464	:	RENUM Zahl1,Zahl2,Zahl3
› SCHNEIDER CPC664,CPC6128	:	RENUM Zahl1,Zahl2,Zahl3
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

RENUMBER

Befehlsart: Kommando

Beschreibung: Das Programm, das sich im Speicher des Rechners befindet, kann anhand dieses Kommandos mit neuen Programmzeilen versehen werden (siehe RENUM).

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	RENUMBER Zahl1,Zahl3,Zahl2
› COMMODORE 128	:	RENUMBER Zahl1,Zahl3,Zahl2
› ENTERPRISE 64,128	:	RENUMBER Zahl2 TO LAST AT Zahl1 STEP Zahl3
MSX-STANDARD	:	RENUM Zahl1,Zahl2,Zahl3
SCHNEIDER CPC464	:	RENUM Zahl1,Zahl2,Zahl3
› SCHNEIDER CPC664,CPC6128	:	RENUM Zahl1,Zahl2,Zahl3
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

RESTORE

Befehlsart: Anweisung

RESTORE

Beschreibung: Als Anweisung in einem Programm – nicht zu verwechseln mit dem Drücken der RESTORE-Taste – bewirkt der Befehl ein Zurücksetzen des READ-Zeigers (siehe unter READ) auf die erste DATA-Information (siehe unter DATA). Der Datenlesevorgang läßt sich also durch RESTORE immer wieder von neuem ausführen.

Beispiel:

```
10 CLS
20 FOR N=1 TO 3
30 READ A$
40 PRINT A$ " ";
50 DATA SCHNEIDER,COMPUTER,BUECHER
60 FOR P= 1 TO 500:NEXT
80 NEXT N
85 PRINT:PRINT:PRINT
90 FOR M= 1 TO 3
100 READ B$
110 PRINT B$ " ";
120 DATA MIT, >RESTORE<,"NOCHMAL:"
130 FOR T= 1 TO 1000:NEXT
140 PRINT
150 NEXT M
160 RESTORE
170 GOTO 20
```

Anmerkung zur Kompatibilität:

Besonderheit:

Beim Sinclair Spectrum hat der Befehl das Format

RESTORE **Zeilennummer.**

Auch in anderen BASIC-Dialekten kann man die **Zeilennummer** angeben (z. B. Schneider CPC). Damit wird der READ-Zeiger auf die erste DATA-Anweisung der angegebenen Zeile gesetzt.

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  RESTORE Zeile
› COMMODORE VC20,64         :  RESTORE
› COMMODORE 16,116,PLUS4    :  RESTORE Zeile
› COMMODORE 128             :  RESTORE Zeile
› ENTERPRISE 64,128         :  RESTORE Zeile
› MSX-STANDARD              :  RESTORE Zeile
› SCHNEIDER CPC464          :  RESTORE Zeile
› SCHNEIDER CPC664,CPC6128  :  RESTORE Zeile
  SINCLAIR ZX81             :
› SINCLAIR SPECTRUM         :  RESTORE Zeile
```

RESUME

Befehlsart: Anweisung

RESUME **Zeilennummer**

Beschreibung: Dieser Befehl ist nur im Zusammenhang mit ON ERROR GOTO (siehe dort) sinnvoll.

Normalerweise bricht der Rechner ein Programm ab, wenn er einen Fehler findet (wie zum Beispiel S$=a). Mit dem Befehl ON ERROR GOTO kann man in eine Zeile springen, in der zum Beispiel die Art des Fehlers ausgegeben wird. Der RESUME-Befehl bewirkt die Rückkehr aus dieser Korrekturzeile zurück ins BASIC-Programm.

Gibt man hinter dem RESUME keine **Zeilennummer** an, so bleibt der Rechner scheinbar "stehen". Der Grund dafür ist, daß mit RESUME zurück in die fehlerhafte Zeile gesprungen wird. Von dort aber wird aufgrund des noch immer vorhandenen Fehlers wieder in die Korrekturzeile gesprungen, und so weiter.

Gibt man dagegen den Befehl RESUME NEXT ein, so erfolgt der Sprung in jene Zeile, die nach der fehlerhaften Zeile folgt, und das Programm wird weiter abgearbeitet.

Bei Angabe einer **Zeilennummer** erfolgt der Sprung zu genau dieser Zeile (z. B. RESUME 1ØØ).

Beispiel:

```
10 ON ERROR GOTO 100
20 INPUT A
30 S$=A
40 PRINT S$
50 END
100 S$=STR$(A)
110 RESUME NEXT
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  RESUME Zeile
> COMMODORE 128             :  RESUME Zeile
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  RESUME Zeile
> SCHNEIDER CPC464          :  RESUME Zeile
> SCHNEIDER CPC664,CPC6128  :  RESUME Zeile
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

RETRY

Befehlsart: Anweisung

RETRY

Beschreibung: Mit dieser Anweisung kann man eine HANDLER-Routine (siehe dort, siehe auch unter WHEN EXCEPTION) verlassen. Der Rechner versucht dann, das Programm an der Stelle fortzusetzen, an der es unterbrochen wurde.

Beispiel:

(siehe unter WHEN EXCEPTION)

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  RETRY
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

RETURN

Befehlsart: Anweisung

RETURN

Beschreibung: Mit RETURN wird vom Ende eines Unterprogrammes ins Hauptprogramm zurückgesprungen, und zwar zu dem Befehl, der nach dem Aufruf des Subprogrammes GOSUB (siehe dort) folgt. Dabei spielt es keine Rolle, ob dieser Befehl in derselben Programmzeile mit dem GOSUB-Befehl oder in der nächsten Zeile erscheint.

Beispiele:

```
10 INPUT A$
20 INPUT "ZAHL DER BUCHSTABEN";A
30 IF A=LEN (A$) THEN GOSUB 1000
```

```
1000 PRINT "GRATULIERE!"
1010 FOR X=1 TO 16
1020 POKE 646,X
1030 NEXT X
1040 RETURN
```

```
10 REM * FORMEL1*
20 PRINT "WIEVIEL KILOMETER IST WAGEN 1 GEFAHREN"
30 INPUT A
40 PRINT "WIEVIEL KILOMETER IST WAGEN 2 GEFAHREN"
50 INPUT B
55 GOSUB 100
60 PRINT:PRINT"ABSTAND ZWISCHEN DEN WAGEN: ":PRINT ABS(A-B); "KILOMETER.":END
100 FOR C=1 TO A:PRINT "*";:NEXT C
105 PRINT"@"
110 FOR D=1 TO B:PRINT "*";:NEXT D
115 PRINT"@"
120 RETURN
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	RETURN
› COMMODORE VC20,64	:	RETURN
› COMMODORE 16,116,PLUS4	:	RETURN
› COMMODORE 128	:	RETURN
› ENTERPRISE 64,128	:	RETURN
› MSX-STANDARD	:	RETURN Zeilennummer
› SCHNEIDER CPC464	:	RETURN
› SCHNEIDER CPC664,CPC6128	:	RETURN
› SINCLAIR ZX81	:	RETURN
› SINCLAIR SPECTRUM	:	RETURN

RGR

Befehlsart: Funktion

RGR(**Zahl**)

Beschreibung: Es wird der augenblickliche Grafik-Modus zurückgegeben, der mit der GRAPHIC-Anweisung (siehe dort) eingestellt worden ist.

Der Wert, den man erhält, liegt zwischen Ø und 5.

Die angegebene **Zahl** ist beliebig wählbar und ohne Einfluß auf die Ausführung der Funktion. Sie darf aber nicht fehlen.

Beispiele:

```
I=RGR(Ø)

PRINT RGR(2)
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  RGR(Zahl)
> COMMODORE 128             :  RGR(Zahl)
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

RIGHT$

Befehlsart: Funktion

RIGHT$(**String,Zahl**)

Beschreibung: Mit dieser Funktion zur Wandlung von Zeichenketten erhält man einen Teilstring des **String**s, der die **Zahl** rechten Zeichen des **String**s enthält.

Beispiel:

```
10 A$="SCHNEIDER COMPUTER BUECHER"
20 PRINT RIGHT$(A$,40)
RUN
SCHNEIDER COMPUTER BUECHER

READY.
```

Anmerkungen zur Kompatibilität: Bei einigen BASIC-Dialekten fehlen die Funktionen LEFT$ (siehe dort) und RIGHT$. Sie sind zu ersetzen durch eine Kombination der Funktionen MID$ oder LEN (siehe dort).

Beispiel:

```
10 A$="SCHNEIDER COMPUTER BUECHER"
20 PRINT MID$(A$,20,7)
RUN
BUECHER

READY.
```

Auch wenn es keine MID$-Funktion gibt, gibt es durchaus Möglichkeiten, um LEFT$ und RIGHT$ zur Wandlung von Zeichenketten zu simulieren.

So erfüllen beim ATARI die Funktion

```
X$ = X$(LEN(X$)-Zahl+1,LEN(X$)
```

und beim Sinclair die Funktion

```
X$=X$(LEN X$-Zahl+1 TO LEN X$)
```

die Funktion

```
RIGHT$(X$,Zahl).
```

Hat der **String** mehr Zeichen, als die Zahl angibt, so wird in allen Fällen der gesamte Textausdruck ausgegeben.

Beispiel:

```
10 A$="SCHNEIDER COMPUTER BUECHER"
20 PRINT RIGHT$(A$,7)
RUN
BUECHER
READY.
```

```
10 A$="SCHNEIDER COMPUTER BUECHER"
20 PRINT MID$(A$,1,30)
RUN
SCHNEIDER COMPUTER BUECHER
READY.
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
› COMMODORE VC20,64         :  RIGHT$(String,Zahl)
› COMMODORE 16,116,PLUS4    :  RIGHT$(String,Zahl)
› COMMODORE 128             :  RIGHT$(String,Zahl)
  ENTERPRISE 64,128         :
› MSX-STANDARD              :  RIGHT$(String,Zahl)
› SCHNEIDER CPC464          :  RIGHT$(String,Zahl)
› SCHNEIDER CPC664,CPC6128  :  RIGHT$(String,Zahl)
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

RLUM

Befehlsart: Funktion

RLUM(Zahl)

Beschreibung: Es wird der Helligkeitswert einer Farbzone zurückgegeben. Diesen Helligkeitswert kann man mit der COLOR-Anweisung (siehe dort) verändern.

Die **Zahl** gibt die Farbzone an, deren Helligkeitswert abgefragt werden soll. Ihr Wert liegt zwischen Ø und 4.

Als Ergebnis erhält man eine Zahl zwischen Ø (stärkste Dunkelheit) und 8 (größte Helligkeit).

Beispiele:

```
PRINT RLUM(1)

A=RLUM(3)
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	RLUM(Zahl)
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

RND

Befehlsart: Funktion

RND(**Zahl**)

Beschreibung: Die Funktion erzeugt eine Zufallszahl zwischen Ø und 1. Die **Zahl** ist 1 oder eine andere, ganze positive Zahl. Sie hat keinen Einfluß auf den zurückgegebenen Wert, darf aber nicht fehlen.

Beispiel:

```
10 PRINT RND(-TI)
20 A=INT(RND(1)*6+1):PRINT"␣"
30 IF A=1 THEN GOSUB 100:GOTO 1000
40 IF A=2 THEN GOSUB 200:GOTO 1000
50 IF A=3 THEN GOSUB 300:GOTO 1000
60 IF A=4 THEN GOSUB 400:GOTO 1000
70 IF A=5 THEN GOSUB 500:GOTO 1000
80 IF A=6 THEN GOSUB 600:GOTO 1000
100 PRINT "┌─┐"
110 PRINT "│ │"
120 PRINT "│●│"
130 PRINT "│ │"
140 PRINT "└─┘"
150 RETURN
200 PRINT "┌─┐"
210 PRINT "│●│"
220 PRINT "│ │"
230 PRINT "│●│"
240 PRINT "└─┘"
250 RETURN
300 PRINT "┌─┐"
310 PRINT "│●│"
320 PRINT "│●│"
330 PRINT "│●│"
340 PRINT "└─┘"
350 RETURN
400 PRINT "┌─┐"
410 PRINT "│●●│"
420 PRINT "│ │"
430 PRINT "│●●│"
440 PRINT "└─┘"
450 RETURN
500 PRINT "┌─┐"
510 PRINT "│●●│"
520 PRINT "│●│"
530 PRINT "│●●│"
540 PRINT "└─┘"
550 RETURN
600 PRINT "┌─┐"
610 PRINT "│●●│"
620 PRINT "│●●│"
630 PRINT "│●●│"
640 PRINT "└─┘"
650 RETURN
1000 INPUT "␣␣␣␣␣␣␣␣␣␣NEUER WURF (J/N)";Q$:IFQ$="J" THEN PRINT"␣":GOTO 20
1010 IF Q$="N" THEN PRINT"␣":END
1020 PRINT "WAS DENN NUN?":GOTO 1000

READY.
```

Das Programm erzeugt die Würfe eines Würfels.

Anmerkung: Wenn die erste Zufallszahl bei den Commodore-Rechnern mit RND(-TI) erzeugt wird, ist sichergestellt, daß diese Zahl jeweils unterschiedlich ist.

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	RND(Zahl)
› COMMODORE VC20,64	:	RND(Zahl)
› COMMODORE 16,116,PLUS4	:	RND(Zahl)
› COMMODORE 128	:	RND(Zahl)
› ENTERPRISE 64,128	:	RND(Zahl)
› MSX-STANDARD	:	RND(Zahl)
› SCHNEIDER CPC464	:	RND(Zahl)
› SCHNEIDER CPC664,CPC6128	:	RND(Zahl)
› SINCLAIR ZX81	:	RND
› SINCLAIR SPECTRUM	:	RND

ROUND

Befehlsart: Anweisung

ROUND(**Zahl 1,Zahl 2**)

Beschreibung: Die **Zahl 1** wird gerundet. Die Anzahl der Stellen nach dem Komma legt man durch **Zahl 2** fest. Dabei bedeutet:

Zahl 2=9	9 Stellen nach dem Komma
Zahl 2=8	8 Stellen nach dem Komma
Zahl 2=2	2 Stellen nach dem Komma
Zahl 2=1	1 Stelle nach dem Komma
Zahl 2=Ø	keine Stelle vor oder nach dem Komma
Zahl 2=-1	1 Stelle vor dem Komma
Zahl 2=-2	2 Stellen vor dem Komma
Zahl 2=-9	9 Stellen vor dem Komma

Rundet man vor dem Komma, werden die fehlenden Stellen mit Nullen aufgefüllt. Wenn wir zum Beispiel die Zahl 123 auf

1 Stelle vor dem Komma runden, wird 12Ø als Ergebnis zurückgegeben.

Beispiel:

```
10 x=987654321
20 FOR i=-9 TO 0
30 PRINT ROUND (x,i)
40 NEXT i
50 x=0.123456789
60 FOR i=1 TO 9
70 PRINT ROUND (x,i)
80 NEXT i
```

Anmerkung zur Kompatibilität: Bei Computern, die diesen Befehl nicht vorweisen, kann das Ergebnis so erzielt werden:

```
INT(/ZAHL1/*10^/ZAHL2/+.5)/10^/ZAHL2/
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
› ENTERPRISE 64,128         :  ROUND(Zahl1,Zahl2)
  MSX-STANDARD              :
› SCHNEIDER CPC464          :  ROUND(Zahl1,Zahl2)
› SCHNEIDER CPC664,CPC6128  :  ROUND(Zahl1,Zahl2)
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

RREG

Befehlsart: Anweisung

RREG **Variable 1,Variable 2,Variable 3,Variable 4**

Beschreibung: Nach Beendigung eines mit SYS (siehe dort) aufgerufenen Maschinenprogramms werden die Registerinhalte des Mikroprozessors den Variablen zugewiesen. Alle Variablen sind Zahlenvariablen.

Variable 1	erhält den Wert des Akkumulators des Prozessors zugewiesen,
Variable 2	den Wert des x-Registers,
Variable 3	den Wert des y-Registers und
Variable 4	den Wert des Statusregisters.

Beispiel:

```
RREG A,X,Y,S
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :
        RREG AVariable1,XVariable2,YVariable3,SVariable4
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

RSPCOLOR

Befehlsart: Funktion

RSPCOLOR(Zahl)

Beschreibung: Die Funktion fragt die Farben für Multicolorsprites (Mehrfarbensprites) ab.
Die **Zahl** gibt an, welche Zusatzfarbe abgefragt werden soll. Man erhält als Ergebnis eine Zahl zwischen 1 und 16. Dabei bedeutet

1	Schwarz	5	Violett	9	Hellbraun	13	Grau
2	Weiß	6	Dunkelgrün	10	Braun	14	Hellgrün
3	Rot	7	Blau	11	Hellgrün	15	Hellblau
4	Grün	8	Gelb	12	Dunkelgrau	16	Hellgau

Die **Zahl** kann die Werte 1 oder 2 annehmen, je nachdem, welche Zusatzfarbe abgefragt wird.

Beispiele:

```
I=RSPCOLOR(1)

PRINT RSPCOLOR(2)
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  RSPCOLOR(Zahl)
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

RSPPOS

Befehlsart: Funktion

RSPPOS(**Zahl 1,Zahl 2**)

Beschreibung: Diese Funktion liefert dem Anwender für ein bestimmtes Sprite die Position und Geschwindigkeit.

Die **Zahl 1** gibt die Nummer des Sprites an, das abgefragt werden soll. Sie ist ganzzahlig, und ihr Wert liegt zwischen 1 und 8.

Die **Zahl 2** gibt an, was abgefragt werden soll. Auch sie ist ganzzahlig und liegt zwischen Ø und 2.

Dabei bedeutet

Zahl 2

Ø	die augenblickliche x-Achse (Abszisse bzw. Spalte) des Sprites.
1	die augenblickliche y-Achse (Ordinate bzw. Zeile) des Sprites.
2	die augenblickliche Geschwindigkeit des Sprites. Der zurückgegebene Wert kann zwischen Ø und 15 liegen.

Beispiele:

```
A=RSPPOS(1,Ø)              (x-Koordinate von Sprite 1)

PRINT RSPPOS(3,2)          (Geschwindigkeit von Sprite 3)
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  RSPPOS(Zahl1,Zahl2)
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

RSPRITE

Befehlsart: Funktion

RSPRITE(**Zahl 1**,**Zahl 2**)

Beschreibung: Die Funktion übermittelt dem Anwender die augenblicklichen Sprite-Eigenschaften des mit **Zahl 1** gekennzeichneten Sprites. Die **Zahl 1** hat einen ganzzahligen Wert zwischen 1 und 8.

Die **Zahl 2** gibt jene Sprite-Eigenschaften an, die abgefragt werden sollen. Der Wert ist ganzzahlig und liegt zwischen Ø und 5.

Zahl 2	bewirkt Ausgabe
Ø	1 wenn das Sprite aktiv ist Ø wenn das Sprite nicht aktiv ist
1	Spritefarbe (Werte zwischen 1 und 16)
2	Ø wenn das Sprite Priorität über den Hintergrund hat 1 wenn das Sprite keine Priorität über den Hintergrund hat
3	1 wenn es eine Spritedehnung in x-Richtung (Spalten) gibt Ø wenn es keine Spritedehnung in x-Richtung gibt
4	1 wenn es eine Spritedehnung in y-Richtung (Zeilen) gibt Ø wenn es keine Spritedehnung in y-Richtung gibt
5	1 wenn Spritemehrfarbenmodus aktiv ist Ø wenn Spritemehrfarbenmodus nicht aktiv ist.

Beispiel:

```
PRINT RSPRITE(8,1)
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  RSPRITE(Zahl1,Zahl2)
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

RTRIM$

Befehlsart: Funktion

RTRIM$(**String**)

Beschreibung: Die Funktion entfernt sämtliche Leerzeichen am Ende eines **Strings**.

Beispiel:

```
PRINT RTRIM$("TEST     ");"Stringende"
TESTStringende
READY
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  RTRIM$(String)
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

RUN

Befehlsart: Kommando

RUN
RUN **Zeilennummer**

Beschreibung: Mit dem Kommando RUN wird ein Programm gestartet, das in den Speicher des Computers geladen wurde. Die **Zeilennummer** gibt an, ab welcher Programmzeile das Programm seine Ausführung beginnen soll.

Steht keine **Zeilennummer** hinter dem Kommando, wie es zumeist der Fall ist, wird das Programm ab seinem Anfang gestartet, also ab der ersten Zeile.

RUN **Zeilennummer** verlangt eine im Programm vorhandene, in Zahlen ausgedrückte **Zeilennummer**. Nicht existierende **Zeilennummern** führen zu Fehlermeldungen.

Beispiel:

```
10 PRINT"*"
20 GOTO 10
RUN 15
?UNDEF'D STATEMENT  ERROR
READY.
```

Variablen sind als **Zeilennummern** hinter RUN nicht erlaubt.

Beispiel:

```
10 PRINT"*"
20 GOTO10
RUNX
?UNDEF'D STATEMENT  ERROR
READY.
```

Den Bindestrich darf man ebenfalls nicht verwenden, auch nicht in der Form, wie wir es beim LIST-Kommando kennen.

Beispiel:

```
10 PRINT"*"
20 GOTO10
RUN -10
?UNDEF'D STATEMENT  ERROR
READY.
```

RUN läßt sich allerdings sehr wohl innerhalb eines Programmes verwenden, etwa wenn das Programm unter definierten Bedingungen neu gestartet werden soll.

Beispiel:

```
30 PRINT "***"
40 FOR K=1 TO 70:NEXT
50 GET A$:IFA$="*" THEN RUN
60 PRINT N
```

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  RUN
› COMMODORE VC20,64         :  RUN Zahl
› COMMODORE 16,116,PLUS4    :  RUN Zahl
› COMMODORE 128             :  RUN Zahl
› ENTERPRISE 64,128         :  RUN Zahl
› MSX-STANDARD              :  RUN Zahl
› SCHNEIDER CPC464          :  RUN Zahl
› SCHNEIDER CPC664,CPC6128  :  RUN Zahl
› SINCLAIR ZX81             :  RUN Zahl
› SINCLAIR SPECTRUM         :  RUN Zahl
```

RUN

Befehlsart: Kommando

RUN **"Programmname"**

Beschreibung: Mit RUN **"Programmname"** wird ein BASIC-Programm von der Diskette oder Kassette geladen und sofort gestartet. Jedes vorher geladene BASIC-Programm wird dabei aus dem Arbeitsspeicher gelöscht (siehe auch START).

Mit diesem Kommando kann man geschützte BASIC-Programme direkt bearbeiten.

Beispiel:

RUN "Hexenmeister"

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  RUN "Programmname"
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
› COMMODORE 128             :  RUN "Programmname",DZahl1,UZahl2
› ENTERPRISE 64,128         :  RUN "Programmname"
  MSX-STANDARD              :
› SCHNEIDER CPC464          :  RUN "Programmname"
› SCHNEIDER CPC664,CPC6128  :  RUN "Programmname"
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

RWINDOW

Befehlsart: Funktion

RWINDOW(**Zahl**)

Beschreibung: Die Funktion gibt die aktuellen Parameter eines Bildschirmfensters wieder, das mit dem WINDOW-Befehl (siehe dort) definiert wurde.

Die **Zahl** gibt an, welche Parameter des WINDOWs abgefragt werden. Sie ist ganzzahlig; ihr Wert liegt zwischen Ø und 2. Dabei bedeutet

Zahl

Ø die Nummer der letzten Zeile des WINDOWs (Ø bis 24).
1 die Nummer der letzten Spalte des WINDOWs (Ø bis 79).
2 die Zahl 4Ø oder 8Ø, je nach dem eingestellten Bildschirm-Modus.

Beispiele:

```
I=RWINDOW(1)

PRINT RWINDOW(2)
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  RWINDOW(Zahl)
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

SAVE

Befehlsart: Kommando

SAVE
SAVE **"Programmname"**
SAVE **"Programmname", Geräteadresse, Zahl**
SAVE **"Programmname", Dateityp, Binärdateiparameter**

Beschreibung: Ein im Hauptspeicher des Rechners befindliches Programm wird in einem Speichermedium (Kassette, Diskette) abgespeichert.

Anmerkung zur Kompatibilität: Das SAVE-Kommando ist in jedem der beschriebenen BASIC-Dialekte vorgesehen. Der Befehl unterscheidet sich allerdings in den einzelnen Dialekten hinsichtlich des Formats (siehe Kompatibilitätsliste).
Wenn zum Beispiel bei einigen Dialekten

SAVE

ohne Parameter eingegeben wird, so wird das augenblicklich im Rechnerspeicher befindliche Programm auf dem Kassettenrecorder abgespeichert (Commodore-Computer usw.). In anderen Dialekten ist die Angabe des **Programmnamens** zwingend vorgeschrieben (Schneider CPC usw.)
Der **Programmname** bezeichnet das abgespeicherte Programm; unter diesem Namen kann man es wieder mit LOAD (siehe dort) in den Rechnerspeicher laden.

Anhand der **Geräteadresse** wird ein bestimmtes Ein-/Ausgabegerät angesprochen, z. B.:

```
SAVE "COMPUTER-ANIMATION",8
```

Die **Zahl** kann den Wert Ø oder den Wert 1 annehmen. Ist der Wert 1, so wird ein Bandende-Zeichen gesetzt. Beim Laden interpretiert der Commodore-Rechner dieses Zeichen als Befehl, den Suchvorgang abzubrechen und eine Fehlermeldung
FILE NOT FOUND ERROR
auszugeben.

Beispiele:

SAVE	Ein Programm ohne Namen wird auf dem Kassettenrecorder abgespeichert.
SAVE "Hyperprogramm"	Ein Programm mit dem Namen "Hyperprogramm" wird auf dem Kassettenrecorder abgespeichert.
SAVE A$	Ein Programm mit dem Programmnamen, der vorher der Stringvariablen A$ übergeben worden ist, wird auf dem Kassettenrecorder abgespeichert.
SAVE "Superprogramm",8	Ein Programm mit dem Namen "Superprogramm" wird auf Diskette gespeichert.
SAVE "Geheimprogramm",1,1	Ein Programm mit dem Namen "Geheimprogramm" wird auf dem Kassettenrecorder abgespeichert. Nach dem Speichervorgang wird ein Bandende-Kennzeichen abgelegt.

(Beispiele für Commodore-Computer)

Das Kommando SAVE kann sowohl im Direktmodus als auch innerhalb eines Programmes verwendet werden:

Beispiel:

```
190 CLS
200 PRINT "WOLLEN SIE DAS PROGRAMM ABSPEICHERN?"
210 A$=INKEY$:IF A$=""THEN GOTO 210
220 IF A$="J" OR A$="j" THEN GOTO 240
230 END
240 PRINT "WOLLEN SIE DAS PROGRAMM SCHUETZEN?"
250 B$=INKEY$:IF B$="" THEN GOTO 250
260 IF B$="J" OR B$= "j" THEN GOTO 280
270 SAVE "HYPERPROGRAMM"
280 SAVE "HYPERPROGRAMM",P
```

In Zeile 28Ø ist bei diesem Beispiel (für den Schneider-CPC) der **Dateityp** angegeben.
Dabei bedeutet

Dateityp

A	Sicherung als ASCII-Datei
P	Sicherung als geschütztes Programm
B	Speicherung als Binärdatei (z. B. Bildschirmbereich)

Im letzten Fall geben die **Binärdateiparameter** die Startspeicheradresse und die Länge an.

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  SAVE "C:Programmname"
                               CSAVE "Programmname"
› COMMODORE VC20,64         :  SAVE "Programmname"
› COMMODORE 16,116,PLUS4    :  SAVE "Programmname"
› COMMODORE 128             :  SAVE "Programmname"
› ENTERPRISE 64,128         :  SAVE "Programmname"
› MSX-STANDARD              :  SAVE "CAS:Programmname"
                               CSAVE "Programmname"
› SCHNEIDER CPC464          :  SAVE "Programmname"
  SCHNEIDER CPC664,CPC6128  :
› SINCLAIR ZX81             :  SAVE "Programmname"
› SINCLAIR SPECTRUM         :  SAVE "Programmname"
```

Programm abspeichern auf eine Diskette

```
› ATARI 600,800,130XE       :  SAVE "D:Programmname"
› COMMODORE VC20,64         :  SAVE "Programmname",Zahl
› COMMODORE 16,116,PLUS4    :  SAVE "Programmname",Zahl
                               DSAVE "Programmname",D0,UZahl
› COMMODORE 128             :  SAVE "Programmname",Zahl
                               DSAVE "Programmname",D0,UZahl
› ENTERPRISE 64,128         :  SAVE "Programmname"
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
› SCHNEIDER CPC664,CPC6128  :  SAVE "Programmname"
  SINCLAIR ZX81             :
› SINCLAIR SPECTRUM         :  LOAD*"M";Laufwerksnummer;"Programmname"
```

SCALE

Befehlsart: Anweisung

SCALE **Zahl 1,Zahl 2,Zahl 3**

Beschreibung: Es wird ein anderer Maßstab für die grafische Darstellung auf dem Bildschirm gewählt. Man kann diese Maßstabsveränderung ein- bzw. ausschalten.

Zahl 1 bestimmt, ob diese "Skalierung" ein- (1) oder ausgeschaltet (Ø) wird. Nach dem Ausschalten ist die ursprüngliche Einteilung des Bildschirms gegeben:

	x	y
hochauflösend	Ø - 319	Ø - 199
mehrfarbig	Ø - 159	Ø - 199

Zahl 2 gibt einen Maßstab für die x-Achse (Abszisse) auf dem Bildschirm an (Spalten). Es wird eine neue Höchstgrenze für den Wert auf x-Achse festgelegt. Dieser Wert kann zwischen dem ursprünglichen Höchstwert und 1Ø23 liegen.

Zahl 3 gibt einen neuen Maßstab für die y-Achse (Ordinate) auf dem Bildschirm (Zeilen) an. Es wird eine neue Höchstgrenze für den Wert auf der y-Achse festgelegt. Dieser Wert kann zwischen 199 und 1Ø23 liegen.

Beispiele:

SCALE Ø
Dieser Ausdruck schaltet die Skalierung ab.

SCALE 1,1ØØØ,1ØØØ

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	SCALE Zahl1,Zahl2,Zahl3
› COMMODORE 128	:	SCALE Zahl1,Zahl2,Zahl3
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SCNCLR

Befehlsart: Kommando

SCNCLR **Zahl**

Beschreibung: Mit diesem Befehl löscht man die einzelnen Grafikbildschirme.

Die **Zahl** gibt an, welchen Grafikschirm man löscht. Sie hat einen ganzzahligen Wert zwischen Ø und 5.

Zahl

Ø	4Ø-Zeichen-Bildschirm wird gelöscht.
1	Der hochauflösende Grafikschirm (32Ø*2ØØ) wird gelöscht.
2	Der hochauflösende Grafik-Text-Bildschirm wird gelöscht.
3	Der Mehrfarben-Grafikschirm (16Ø*2ØØ) wird gelöscht.
4	Der Mehrfarben-Grafik-Text-Bildschirm wird gelöscht.
5	Der 8Ø-Zeichen-Bildschirm wird gelöscht.

Beispiel:

```
SCNCLR 1
```

Anmerkung: Bei einem Bildschirm, der durch den GRAPHIC-Befehl (siehe dort) eingestellt ist, kann man mit SCNCLR ohne Angabe der **Zahl** löschen

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	SCNCLR Zahl
› COMMODORE 128	:	SCNCLR Zahl
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SCRATCH

Befehlsart: Kommando

SCRATCH "**Programmname**",D**Zahl 1**,U**Zahl 2**

Beschreibung: Eine Datei mit dem **Programmnamen** wird auf einer Diskette gelöscht.

Der **Programmname** darf bis zu 16 Zeichen lang sein. Gibt man nur die ersten Buchstaben des **Programmnamens** und als letztes Zeichen ein * ein, werden alle Dateien, deren Name mit diesen Buchstaben beginnt, gelöscht.

Die **Zahl 1** gibt die Laufwerksnummer der Floppy an, auf der die Datei gelöscht werden soll. Sie beträgt entweder Ø oder 1. Man muß sie nicht in jedem Falle angeben.

Die **Zahl 2** ist nur dann anzugeben, wenn mehrere Diskettenlaufwerke verwendet werden. Sie gibt dann die Nummer des Laufwerkes an, die normalerweise zwischen 7 und 1Ø liegt. Sie kann aber auch Werte zwischen 4 und 15 annehmen.

Hat man den Befehl eingegeben und mit RETURN bestätigt, erscheint auf dem Bildschirm die Frage

ARE YOU SURE? (Sind Sie sicher?).

Bei Eingabe von Y wird nun die Datei gelöscht, andernfalls wird der Befehl ignoriert.

Am Ende des Löschvorgangs wird dieser durch folgende Meldung bestätigt:

Ø1,FILES SCRATCHED,**Zahl**,ØØ

Die **Zahl** steht für die Anzahl der gelöschten Dateien.

Beispiele:

```
SCRATCH "APFEL",DØ,U9
SCRATCH "DE*",DØ
SCRATCH "Versuch"
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE          :
  COMMODORE VC20,64            :  OPEN1,8,Zahl2,"S:Programmname"
> COMMODORE 16,116,PLUS4       :  SCRATCH "Programmname",DZahl1,UZahl2
> COMMODORE 128                :  SCRATCH "Programmname",DZahl1,UZahl2
  ENTERPRISE 64,128            :
  MSX-STANDARD                 :
  SCHNEIDER CPC464             :
  SCHNEIDER CPC664,CPC6128     :
  SINCLAIR ZX81                :
  SINCLAIR SPECTRUM            :
```

SCREEN

Befehlsart: Kommando

SCREEN **Zahl 1,Zahl 2,Zahl 3,Zahl 4,Zahl 5**

Beschreibung: Das SCREEN-Kommando dient zum Einschalten folgender Parametergrößen:

Zahl 1	Betrieb	Bildschirmmodus
Zahl 2	Sprite	Zähler der Sprite-Größe
Zahl 3	Klick	Tastaturklick Ein/Aus
Zahl 4	Baud	Übertragungsgeschwindigkeit zum Kassettenrecorder
Zahl 5	Druck	Angabe des Druckertyps

Beispiel:

SCREEN ,,,2
(Die Baud-Zahl des Kassetten-Interfaces wird auf 2400 eingestellt.)

SCREEN 2,3
(Die hochauflösende Grafik mit einer Sprite-Größe von 16 x 16 Punkten wird aufgerufen.)

Anmerkung: Es gibt sehr viele Möglichkeiten für die Angaben der Parametergrößen. Es ist daher meist von Nutzen,

die jeweiligen Handbücher der MSX-Computer zu Rate zu ziehen.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  siehe Beschreibung
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

SCREEN$

Befehlsart: Funktion/Parameter

SCREEN$(**Zahl 1**,**Zahl 2**)
LOAD "**Name**" SCREEN$
SAVE "**Name**" SCREEN$

Beschreibung: Mit der Funktion

SCREEN$(**Zahl 1**,**Zahl 2**)

kann man einer Stringvariablen dasjenige Zeichen zuweisen, das sich an der Stelle befindet, die durch **Zahl 1** (= Wert auf der x-Achse bzw. Spalte) und **Zahl 2** (= Wert auf der y-Achse bzw. Zeile) gekennzeichnet ist.

Anhand der Kommandos

LOAD "**Name**" SCREEN$
SAVE "**Name**" SCREEN$

kann man ein Bildschirm-Bild laden oder abspeichern (siehe auch unter LOAD und SAVE). Das entsprechende Bild wird unter dem **Namen** abgespeichert bzw. aufgerufen.

Beispiel:

```
LET A$=SCREEN$(Ø,Ø)
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
> SINCLAIR SPECTRUM         :  siehe Beschreibung
```

SCROLL

Befehlsart: Kommando

SCROLL

Beschreibung: Der Bildschirminhalt wird um eine Zeile nach oben gerollt. Die oberste Zeile geht verloren; unten wird eine neue Zeile angefügt.

Beispiel:

```
SCROLL
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
> SINCLAIR ZX81             :  SCROLL
  SINCLAIR SPECTRUM         :
```

SEC

Befehlsart: Funktion

SEC(**Zahl**)

Beschreibung: Die Funktion gibt den SEKANS der **Zahl** wieder. Die **Zahl** als Argument des SEKANS darf nicht die Werte (2k+1)*9Ø Grad bzw. (2k+1)*PI/2 annehmen (k steht dabei für eine beliebige ganze Zahl).

Beispiele:

```
PRINT SEC(1)
Y=SEC(X)
```

Anmerkung: Die Funktion sollte nur verwendet werden, wenn man mit den Winkelfunktionen vertraut ist.

Anmerkung zur Kompatibilität: Dieser Befehl kann wie folgt simuliert werden:

SEC(**Zahl**)

entspricht dabei

1/SIN(**Zahl**)

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	SEC(Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SELECT

Befehlsart: Anweisung

SELECT **String**

Beschreibung: Mit dieser Anweisung läßt sich eine durch den **String** bezeichnete Variable auf verschiedene Bedingungen hin untersuchen.

Beispiel:

```
10 INPUT NAME$
20 LET NAME$=LCASE$(NAME$)
30 SELECT CASE NAME$
40 CASE "Peter"
50   PRINT "Der Name gefaellt mir gut."
60 CASE "Georg"
70   PRINT "Das ist ein huebscher Name."
80 CASE "Thomas"
90   LET NAME$="Tom"
100 CASE ELSE
110   PRINT "Ich heisse Enterprise!"
120 END SELECT
130 PRINT "Du heisst, glaube ich, " NAME$
```

Zeile 3Ø	Angabe der Variablen, die untersucht werden soll.
Zeile 4Ø	Wenn die Variable die Bedingung Name$= "Peter" erfüllt, wird in Zeile 5Ø die PRINT-Anweisung ausgeführt.
Zeile 6Ø,8Ø	wie Zeile 4Ø.
Zeile 1ØØ	Alle anderen Möglichkeiten, die nicht der Bedingung in Zeile 4Ø, 6Ø oder 8Ø entsprechen, werden nun abgearbeitet.
Zeile 12Ø	Ende des SELECT-Blocks. Sobald eine der Bedingungen erfüllt wurde, wird dorthin verzweigt.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  siehe Beschreibung
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

SET

Befehlsart: Kommando

SET **Option**

Beschreibung: Durch diesen Befehl kann man bestimmte Rechnereigenschaften, wie zum Beispiel die Übertragungsgeschwindigkeit der seriellen Schnittstelle, festlegen. Er sollte nur von schon etwas erfahreneren Programmierern verwendet werden, da es leicht zu Fehlern kommt.

Mit der **Option** gibt man die Rechnereigenschaft an, die neu festgelegt wird. Zumeist geschieht das in Verbindung mit einer Zahl, die den Wert für die Festlegung bestimmt.

Optionen sind

ATTRIBUTES
BEAM
BIAS
BORDER
CHARACTER
COLOUR
CURSOR
DEFAULT CHANNEL
EDITOR BUFFER
EDITOR KEY
EDITOR VIDEO
FAST SAVE
INK
INTERRUPT
KEY CLICK
KEY DELAY
KEY RATE
LINE MODE
LINE STYLE
NET NUMBER
PALETTE
PAPER
REM 1
REM 2
SCROLL
SERIAL BAUD
SERIAL FORMAT
SOUND BUFFER
SPEAKER
STATUS
TAPE LEVEL
TAPE SOUND
TIMER
VIDEO COLOR
VIDEO MODE
VIDEO X
VIDEO Y

Die genauen Verwendungen dieser **Option**en sind jeweils im Handbuch des ENTERPRISE nachzulesen. Man wird hier erst mit ein wenig Übung im Programmieren befriedigende Ergebnisse erzielen.

Beispiel:

SET SERIAL BAUD 6

Die Übertragungsgeschwindigkeit der seriellen Schnittstelle beträgt nun 3ØØ Baud. Dies ist zum Beispiel üblich bei Telefonmodems.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  SET OPTION Parameter
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :  FORMAT OPTION Parameter
```

SETCOLOR

Befehlsart: Anweisung

SETCOLOR **Zahl 1,Zahl 2,Zahl 3**

Beschreibung: Mit diesem Befehl kann man die 5 Farbregister des Atari nach eigenen Wünschen verändern. Es stehen dafür 246 Kombinationen zur Verfügung.

Zahl 1 bezeichnet das gewünschte Farbregister.
Zahl 2 wählt zwischen 16 verschiedenen Farbtönen.
Zahl 3 gibt den Helligkeitswert des gewählten Farbtons wieder.

Die Anzahl der darstellbaren Farben hängt von der zuvor gewählten Auflösung des Grafikbildschirms ab.

Beispiel:
```
10 FOR M=0 TO 15
20 FOR L=0 TO 15
30 SETCOLOR 4,M,L
40 FOR Z=1 TO 100: NEXT
50 NEXT L:NEXT M
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	SETCOLOR Zahl1,Zahl2,Zahl3
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SGN

Befehlsart: Funktion

SGN(**Zahl**)

Beschreibung: Die Funktion gibt das Vorzeichen einer **Zahl** wieder.

Als Ergebnis erhält man

-1	wenn die **Zahl** kleiner Null ist.
Ø	wenn die **Zahl** gleich Null ist.
1	wenn die **Zahl** größer Null ist.

Beispiel:

```
? SGN(-123)
-1
READY
```

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  SGN(Zahl)
› COMMODORE VC20,64         :  SGN(Zahl)
› COMMODORE 16,116,PLUS4    :  SGN(Zahl)
› COMMODORE 128             :  SGN(Zahl)
› ENTERPRISE 64,128         :  SGN(Zahl)
› MSX-STANDARD              :  SGN(Zahl)
› SCHNEIDER CPC464          :  SGN(Zahl)
› SCHNEIDER CPC664,CPC6128  :  SGN(Zahl)
› SINCLAIR ZX81             :  SGN Zahl
› SINCLAIR SPECTRUM         :  SGN Zahl
```

SIN

Befehlsart: Funktion

SIN(**Zahl**)

Beschreibung: Das Ergebnis der mathematischen Funktion ist die Wiedergabe der Winkelfunktion SINUS.
Die **Zahl** ist dabei der Winkel. Er wird im Bogenmaß angegeben. Das Ergebnis ist ein Wert zwischen -1 und 1.

Beispiel:

```
PRINT SIN(1)
 Ø.8414'7Ø985
READY

PRINT SIN(Ø)
 Ø
READY
```

Anmerkung: Die Funktion sollte nur verwenden, wer mit den Winkelfunktionen vertraut ist. Um das Ergebnis in Grad wiederzugeben, muß es mit 57.29578 malgenommen werden.

Anmerkung zur Kompatibilität: Bei ZX81, ZX Spectrum und Sinclair QL hat der Befehl das Format
SIN **Zahl**

Kompatibilitätsliste:

› ATARI 600,800,130XE	: SIN(Zahl)
› COMMODORE VC20,64	: SIN(Zahl)
› COMMODORE 16,116,PLUS4	: SIN(Zahl)
› COMMODORE 128	: SIN(Zahl)
› ENTERPRISE 64,128	: SIN(Zahl)
› MSX-STANDARD	: SIN(Zahl)
› SCHNEIDER CPC464	: SIN(Zahl)
› SCHNEIDER CPC664,CPC6128	: SIN(Zahl)
› SINCLAIR ZX81	: SIN Zahl
› SINCLAIR SPECTRUM	: SIN Zahl

SINH

Befehlsart: Funktion

SINH(**Zahl**)

Beschreibung: Die Funktion gibt den SINUSHYPERBOLICUS einer **Zahl** wieder.

Beispiele:

```
PRINT SINH(1)

A=SINH(B)
```

Anmerkung: Die Funktion sollte nur verwendet werden, wenn man mit den Winkelfunktionen vertraut ist.

Anmerkung zur Kompatibilität: Bei Rechnern, die über diesen Befehl nicht verfügen, kann er wie folgt simuliert werden:

SINH(**Zahl**)

entspricht

(EXP(**Zahl**)-EXP(-**Zahl**))/2

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  SINH(Zahl)
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

SIZE

Befehlsart: Funktion

SIZE(**Variablenname**)
SIZE(**Variablenname**,**Zahl**)

Beschreibung: Die Funktion gibt die Anzahl der Elemente in einem durch DIM, NUMERIC oder STRING (siehe jeweils dort) definierten Feld wieder.

Der **Variablenname** bezeichnet die Variable, die abgefragt werden soll.

Die **Zahl** gibt die Dimension eines Feldes an.

SIZE(**Variablenname**) fragt nach den Elementen eines eindimensionalen Feldes.

SIZE(**Variablenname**,3) fragt nach den Elementen eines mehrdimensionalen Feldes, im Höchstfall einem dreidimen-

sionalen Feld. Die **Zahl** bei einem dreidimensionalen Feld kann die Werte 1, 2 oder 3 annehmen.

Beispiel:

```
PRINT SIZE(A$)
```

Kompatibilitätsliste:

ATARI 600,800,130XE :	
COMMODORE VC20,64 :	
COMMODORE 16,116,PLUS4 :	
COMMODORE 128 :	
ENTERPRISE 64,128 :	SIZE(Variable,Zahl)
MSX-STANDARD :	
SCHNEIDER CPC464 :	
SCHNEIDER CPC664,CPC6128 :	
SINCLAIR ZX81 :	
SINCLAIR SPECTRUM :	

SLEEP

Befehlsart: Kommando

SLEEP **Zahl**

Beschreibung: Die Programmausführung wird vom Anwender für eine von ihm bestimmte Zeit angehalten.

Die **Zahl** gibt dabei die Zeit an, für die die Programmausführung unterbrochen wird. Sie ist ganzzahlig und liegt zwischen 1 und 65535. Die **Zahl** gibt die Zeit in Sekunden an. Das heißt, mit dem SLEEP-Befehl kann man die Programmausführung für einen Zeitraum zwischen 1 Sekunde und rund 18 Stunden unterbrechen.

Beispiel:

```
SLEEP 20        Das Programm wird für 20 Sekunden angehalten.
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
› COMMODORE 128	:	SLEEP Zahl
ENTERPRISE 64,128	:	WAIT DELAY Zahl
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SLOW

Befehlsart: Kommando

SLOW

Beschreibung: Dieser Befehl schaltet den Rechner vom schnellen Verarbeitungsmodus (2 MHz) auf den langsamen (1 MHz). Der Computer arbeitet nun wieder in seiner normalen Verarbeitungsgeschwindigkeit, die beim Einschalten des Rechners wirksam ist.

In diesem Modus kann man sowohl mit einem 8Ø-Zeichen-Monitor als auch mit einem 4Ø-Zeichen-Monitor oder einem normalen Fernseher arbeiten.

Der Befehl mit der entgegengesetzten Wirkung heißt FAST (siehe dort). Er schaltet den Rechner vom langsamen Verarbeitungsmodus (1 MHz) auf den schnellen (2 MHz) um.

Beispiel: SLOW

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
› COMMODORE 128	:	SLOW
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SOUND

Befehlsart: Anweisung

SOUND **Zahl 1,Zahl 2,Zahl 3,Zahl 4,Zahl 5,Zahl 6,Zahl 7**

Beschreibung: Die Anweisung erzeugt einen Ton. **Zahl 1** bezeichnet den Kanal-Status. Einer oder mehrere der drei SOUND-Kanäle des CPC werden angesprochen. Zwar wird die Zahl als ganzzahliger dezimaler Wert zwischen 1 und 255 eingegeben, aber als 8-Bit-Darstellung steuert sie die Anweisung:

Zahl	Bit	Anweisung
1	Ø	SOUND nach Kanal A
2	1	SOUND nach Kanal B
4	2	SOUND nach Kanal C
8	3	Rendezvous mit Kanal A
16	4	Rendezvous mit Kanal B
32	5	Rendezvous mit Kanal C
64	6	Ton halten
128	7	Abbrechen der Töne (flush)

Die **Zahl**en können miteinander kombiniert werden. So bedeutet:

3 (=2+1)	SOUND nach Kanal A und B
7 (=3+2+1)	SOUND nach Kanal A,B und D
97 (=64+32+2)	SOUND nach Kanal B, Rendezvous mit Kanal C, Ton halten.

Zahl 2 kennzeichnet die Frequenz des Tones. Der Wert für **Zahl 2** liegt dabei zwischen Ø und 4Ø95. Sie ist für die Tonhöhe verantwortlich, wobei der niedrigste Wert den höchsten Ton und der größte Wert den tiefsten Ton erzeugt.

Die Angabe von **Zahl 1** und **Zahl 2** ist zwingend notwendig. Mit ihnen kann man bereits ein sinnvolles Format des SOUND-Befehles bilden, weil bei den übrigen **Zahl**en jeweils

ein Standardwert angenommen wird, falls sie nicht angegeben werden.

Beispiel:

```
10 FOR X=1 TO 200
20 SOUND 5,X
30 NEXT X
```

Zahl 3 bestimmt die Dauer des Tons. Ihr Wert kann zwischen -132768 und 32767 liegen. Ist die Zahl nicht angegeben, beträgt der Wert 2Ø.

Falls die eingegebene Zahl positiv ist, beträgt die Tondauer das durch **Zahl** 3 bestimmte Vielfache von 1/1ØØ Sekunde. Bei **Zahl 3** = 1ØØ beträgt die Dauer des Tones also 1 Sekunde.

Ist die Zahl hingegen negativ, ändert sich die Tondauer zu einem durch die **Zahl 3** angegebenen Vielfachen der in **Zahl 5** niedergelegten Lautstärken-Hüllkurve des Tons.

Zahl 4 gibt die Lautstärke des Tons wieder. Sie liegt zwischen Ø und 7, wenn in **Zahl 5** kein Wert für die Lautstärken-Hüllkurve angegeben ist. Andernfalls kann sie Werte zwischen Ø und 15 annehmen. Ist kein Wert angegeben, so nimmt der Rechner automatisch den Wert 4 an, sofern kein Wert für die Lautstärken-Hüllkurve angegeben wurde, und erhält den Wert 12, falls ein Wert vorhanden ist.

Zahl 5 bestimmt die Lautstärken-Hüllkurve. Sie wurde vorher mit dem Befehl ENV (siehe dort) bestimmt.

Zahl 6 gibt die Ton-Hüllkurve an. Sie besitzt einen ganzzahligen Wert zwischen Ø und 15. Dieser Wert bearbeitet eine vorgegebene Hüllkurve, die mit dem Befehl ENT (siehe dort) definiert wurde. Wenn **Zahl 6** nicht angegeben wird, nimmt der Rechner automatisch die Ø an.

Zahl 7 bestimmt die Geräusch-Periode eines Geräusches, das zu SOUND gefügt wird. Das Geräusch wird nicht erzeugt, sobald eine **Ø** als **Zahl 7** eingegeben wird oder wenn diese fehlt.

Beispiel:

```
10 FOR X= 10 TO 50
20 SOUND 1,10,10,15
30 SOUND 1,X,10,7
40 SOUND 2,50,10,15
50 NEXT X
```

Anmerkungen zur Kompatibilität: Die Parameter, die dem SOUND-Befehl folgen, sind in den verschiedenen BASIC-Dialekten unterschiedlich. Die Tonhöhe wird gelegentlich auch in Anlehnung an das traditionelle Notensystem angegeben. Im MSX-BASIC wird mit

SOUND **Registernummer,Wert**

der **Wert** ins Register mit der **Registernummer** geschrieben.

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	SOUND Parameterliste
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	SOUND Parameterliste
› COMMODORE 128	:	SOUND Parameterliste
› ENTERPRISE 64,128	:	SOUND Parameterliste
› MSX-STANDARD	:	SOUND Parameterliste
› SCHNEIDER CPC464	:	SOUND Parameterliste
› SCHNEIDER CPC664,CPC6128	:	SOUND Parameterliste
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SPACE$

Befehlsart: Anweisung

SPACE$(**Zahl**)

Beschreibung: Ein Textausdruck wird mit einer Anzahl von Leerzeichen gefüllt und ausgegeben. Die Anzahl der Leerzeichen wird durch die **Zahl** angegeben.

Beispiel:

```
10 CLS
20 PRINT "Wer";:GOSUB 140
30 PRINT "spielt";:GOSUB 140
40 PRINT "da mit";:GOSUB 140
50 PRINT "meiner";:GOSUB 140
60 PRINT "Tastatur?";:GOSUB 140
70 INPUT na$
80 CLS
90 PRINT "Ach so!";:GOSUB 140
100 PRINT "Du bist";:GOSUB 140
110 PRINT "es wieder,";:GOSUB 140
120 PRINT na$;".":GOSUB 140
130 END
140 PRINT SPACE$(100)
150 RETURN
```

Anmerkung zur Kompatibilität: Bei Rechnern, deren BASIC nicht über diesen Befehl verfügt, läßt er sich mit SPC und TAB (siehe jeweils dort) simulieren.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  SPACE$(Zahl)
> SCHNEIDER CPC464          :  SPACE$(Zahl)
> SCHNEIDER CPC664,CPC6128  :  SPACE$(Zahl)
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

SPC

Befehlsart: Funktion

SPC (**Zahl**)

Beschreibung: Dieser Befehl ist nur in Verbindung mit dem PRINT-Befehl (siehe PRINT) sinnvoll. Dieser erzeugt Leerstellen entsprechend der durch die **Zahl** angegebenen Anzahl.

Ist die **Zahl** kleiner 1, so wird keine Leerstelle ausgegeben. Ist die **Zahl** größer als die maximal zulässige 255, so wird diese dementsprechend gekürzt.

SPC ist sehr ähnlich dem SPACE-Befehl (siehe SPACE), nur daß SPC auch in der Stringbearbeitung eingesetzt werden kann.

Beispiel:

```
PRINT SPC(4);"  "
READY
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
› COMMODORE VC20,64	:	SPC(Zahl)
› COMMODORE 16,116,PLUS4	:	SPC(Zahl)
› COMMODORE 128	:	SPC(Zahl)
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	SPC(Zahl)
› SCHNEIDER CPC464	:	SPC(Zahl)
› SCHNEIDER CPC664,CPC6128	:	SPC(Zahl)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SPEED INK

Befehlsart: Kommando

SPEED INK **Zahl 1,Zahl 2**

Beschreibung: Durch dieses Kommando kann man die Blinkfrequenz festlegen, falls ein Farbwechsel durch den INK- (siehe dort) und durch den BORDER-Befehl (siehe BORDER) stattfindet.

Die **Zahl** gibt die Anzeigedauer der ersten Farbe an, die **Zahl 2** die der zweiten Farbe. Dabei sind diese Zahlen Faktoren von Ø.Ø2 sek., d.h., Anzeigedauer=Zahl*Ø.Ø2 Sekunden.

Beispiel:

```
10 AFTER 50 GOSUB 100
20 BORDER 0,13
30 CLS:i=0.5
40 GOTO 40
100 i=i*2
110 SPEED INK i,i
120 IF i<50 THEN RETURN
130 END
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	SPEED INK Zahl1,Zahl2
› SCHNEIDER CPC664,CPC6128	:	SPEED INK Zahl1,Zahl2
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SPEED KEY

Befehlsart: Kommando

SPEED KEY **Zahl 1,Zahl 2**

Beschreibung: Mit diesem Kommando kann man die Tastaturabfrage beeinflussen.

Mit dem Befehl KEY DEF (siehe dort) kann man festlegen, ob eine Taste eine Wiederholfunktion besitzt oder nicht. Ein Teil der Tasten hat von vornherein Wiederholfunktion, wie etwa die Leertaste.

Wiederholfunktion bedeutet, daß man, solange die Taste gedrückt wird, Zeichen auf dem Bildschirm erhält, bzw. Zeichen in den Tastaturpuffer geschrieben werden.

Durch das Kommando SPEED KEY kann man diese Wiederholfunktion beeinflussen.

Mit der **Zahl 1** kann man die Zeit verändern, ab der die Wiederholfunktion nach dem Tastendruck einsetzt.

Mit der **Zahl 2** kann man festlegen, in welchen Zeitabständen die Taste wiederholt wird.

Dabei werden die Zeiten wie folgt festgelegt:

Zeit=Zahl*Ø.Ø2 Sekunden.

Die Zahlen dürfen dabei nicht außerhalb des Bereiches von 1 und 255 liegen.

Beispiel: SPEED KEY 1,1

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	SET KEY DELAY Zahl1 SET KEY RATE Zahl2
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	SPEED KEY Zahl1,Zahl2
› SCHNEIDER CPC664,CPC6128	:	SPEED KEY Zahl1,Zahl2
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SPEED WRITE

Befehlsart: Kommando

SPEED WRITE **Zahl**

Beschreibung: Mit Hilfe dieses Kommandos kann man die Datenübertragungsgeschwindigkeit für die Ausgabe auf den Datenrekorder festlegen.

Man hat dabei nur 2 Möglichkeiten:

Zahl=Ø	Geschwindigkeit=1ØØØ BAUD
Zahl=1	Geschwindigkeit=2ØØØ BAUD

BAUD bedeutet normalerweise BIT/Sekunde, das würde bei 1ØØØ BAUD bedeuten, daß 125 Zeichen (=BYTES) pro Sekunde auf den Datenrekorder übertragen werden.

Wird eine Datei von der Kassette gelesen, schaltet der CPC464 automatisch auf die richtige Lesegeschwindigkeit der Kassette um.

Beispiel:

SPEED WRITE Ø

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	SPEED WRITE Zahl
› SCHNEIDER CPC664,CPC6128	:	SPEED WRITE Zahl
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SPEEK

Befehlsart: Funktion

SPEEK(**Segment,Adresse**)

Beschreibung: Die Funktion wirkt wie der PEEK-Befehl (siehe dort); man kann allerdings einen größeren Adreßbereich ansprechen. Das **Segment** legt hierfür den Speicherblock fest, der aus je 64 KByte besteht.

Beispiel:

```
PRINT SPEEK(2,3ØØØØ)
```

Anmerkung zur Kompatibilität: Beim Commodore C128 wird die Wirkung dieser Funktion mit dem BANK-Befehl (siehe dort) und der gewohnten PEEK-Funktion erreicht.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  SPEEK(Segment,Adresse)
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

SPOKE

Befehlsart: Anweisung

SPOKE **Segment,Adresse,Zahl**

Beschreibung: Dieser Befehl ähnelt dem POKE-Befehl (siehe dort). Allerdings lassen sich damit mehr als 65536 Speicherzellen ansprechen.
Das **Segment** legt hierfür einen neuen Adreßbereich fest.

Beispiel:

```
SPOKE 1,1Ø24,144
```

Anmerkung zur Kompatibilität: Der Adreßbereich wird beim Commodore C128 mit dem BANK-Befehl (siehe dort) festgelegt. Damit können mehr als 64 KByte angesprochen werden. Beim Enterprise sind es bis 4Ø96 KByte, also 4 Megabyte.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	BANK Segment:POKE Adresse,Zahl
> ENTERPRISE 64,128	:	SPOKE Segment,Adresse,Zahl
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SPRCOLOR

Befehlsart: Anweisung

SPRCOLOR **Zahl 1,Zahl 2**

Beschreibung: Der Anwender kann mit Hilfe dieser Anweisung die Farben für Multicolorsprites (Mehrfarbensprites) festlegen. Diese Farben kann man mit der RSPCOLOR-Funktion (siehe dort) abfragen.

Zahl 1 legt die erste Zusatzfarbe fest, **Zahl 2** die zweite. Die beiden Zahlen können Werte von 1 bis 16 annehmen.

Dabei bedeuten

1 Schwarz	2 Weiß	3 Rot	4 Grün
5 Violett	6 Dunkelgrün	7 Blau	8 Gelb
9 Hellbraun	10 Braun	11 Rosa	12 Dunkelgrau
13 Grau	14 Hellgrün	15 Hellblau	16 Hellgrau

Beispiel:

```
SPRCOLOR 2,3
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
› COMMODORE 128	:	SPRCOLOR Zahl1,Zahl2
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SPRDEF

Befehlsart: Kommando

SPRDEF

Beschreibung: Der im Commodore 128 integrierte Sprite-Editor wird aufgerufen. Folgende Befehle stehen nun zur Verfügung:

1 - 8	Spritenummer 1 bis 8
A	Automatische Wiederholung für die Blocktasten 1 bis 4 ein/aus
CSR-Tasten	Steuerung des Cursors im Spriteraster
RETURN	Neue Zeile im Sprite-Editor
SHIFT & RETURN	Speicherung des entworfenen Sprites. Ein anschließendes weiteres RETURN führt in den BASIC-Modus zurück.
HOME	Der Cursor wird ins linke obere Eck gebracht
CLR	Löschen eines Sprites
1 - 4	Setzen eines Farbpunktes (1 = Hintergrund, 2 = Vordergrund, 3 = erste Farbe, 4 = zweite Farbe)

Beispiel:

SPRDEF

Kompatibilitätsliste:

	ATARI 600,800,130XE	:
	COMMODORE VC20,64	:
	COMMODORE 16,116,PLUS4	:
›	COMMODORE 128	: SPRDEF
	ENTERPRISE 64,128	:
	MSX-STANDARD	:
	SCHNEIDER CPC464	:
	SCHNEIDER CPC664,CPC6128	:
	SINCLAIR ZX81	:
	SINCLAIR SPECTRUM	:

SPRITE

Befehlsart: Anweisung

SPRITE **Zahl 1,Zahl 2,Zahl 3,Zahl 4,Zahl 5,Zahl 6,Zahl 7**

Beschreibung: Mit dieser Anweisung kann der Anwender die Sprite-Attribute festlegen. Insgesamt sechs Attribute lassen sich beeinflussen. Die Werte von **Zahl 2** bis **Zahl 7** können nach Bedarf verwendet werden oder entfallen.

Zahl 1	gibt an	Sprite-Nummer (Werte 1 bis 8)
Zahl 2	gibt an	Sprite aktiviert (1) oder nicht (Ø)
Zahl 3	gibt an	Sprite-Grundfarbe (Werte 1 bis 16)
Zahl 4	gibt an	Sprite Vorrang vor Zeichen (Ø) oder nicht (1)
Zahl 5	gibt an	Sprite-Dehnung in x-Richtung (1) oder nicht (Ø)
Zahl 6	gibt an	Sprite-Dehnung in y-Richtung (1) oder nicht (Ø)
Zahl 7	gibt an	Sprite im hochauflösenden Modus (Ø) oder im Mehrfarbenmodus (1)

Bei nicht eingegebenen Werten bleibt der jeweils vorher gültige Wert erhalten.

Beispiele: SPRITE 1,1

SPRITE 2,1,1,Ø,1,1,1

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :
      SPRITE Zahl1,Zahl2,Zahl3,Zahl4,Zahl5,Zahl6,Zahl7
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

SPRITE$

Befehlsart: Anweisung

SPRITE$ (**Zahl**)

Beschreibung: Mit Hilfe dieses Befehls wird ein Sprite-Muster definiert.

Ein Sprite ist ein aus Punkten (Pixel) zusammengesetztes Muster. Es kann beim MSX-BASIC zwei Größen-Maßstäbe annehmen:
8 x 8 Punkte
oder
16 x 16 Punkte.

Mit der **Zahl** wird die Nummer des Sprites angegeben.

Da sich das Sprite über den Bildschirm frei bewegen läßt, kann man auf diese Weise insbesondere Spiele und optische Effekte programmieren.

Beispiel:

(siehe unter SPRITE ON/OFF oder PUT SPRITE.)

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	SPRITE$(Zahl)
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SPRITE ON/OFF

Befehlsart: Anweisung

SPRITE ON
SPRITE OFF
SPRITE STOP

Beschreibung: Sollen sich zwei Sprites berühren und ist vorher mit ON SPRITE GOSUB (siehe dort) ein Sprungbefehl definiert worden, dient dieser Befehl zur Durchführung verschiedener Optionen. Anhand von

SPRITE ON

wird die Unterbrechung gültig gemacht, und es wird in ein durch GOSUB bezeichnetes Unterprogramm gesprungen.

SPRITE OFF

macht eine Unterbrechung durch zwei sich berührende Sprites ungültig. Mit

SPRITE STOP

wird das Programm angehalten.

Beispiel:

```
10 SCREEN 2
20 SPRITE$(0)=CHR$(&C3)+CHR$(&DB)+CHR$(&18)+CHR$(H18)+CHR$(&DB)+CHR$($HC3)+CHR$(
&7E)+CHR$(&3C)
30 ON SPRITE GOSUB 100
40 SPRITE ON
50 FOR K=0 TO 245
60 PUT SPRITE 0,(X,100),15,0
70 PUT SPRITE 1,(255-X,100),10,0
80 NEXT X
90 END
100 SPRITE OFF:BEEP
110 SPRITE ON:RETURN
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  SPRITE ON      /     SPRITE OFF
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

SPRSAV

Befehlsart: Anweisung

SPRSAV **Zahl,Variablenname**
SPRSAV **Variablenname,Zahl**

Beschreibung: Einem String wird mit diesem Befehl der Inhalt eines Sprites zugewiesen, oder, im zweiten Format des Befehls, wird einem Sprite der Inhalt eines Strings zugewiesen.

Die **Zahl** gibt die Spritenummer an (1 - 8 sind als Werte möglich).

Der **Variablenname** bezeichnet eine Stringvariable, die die Spriteinformation enthält oder der sie übermittelt wird.

Beispiele:

SPRSAV 2,A$

Der Variablen A$ wird das Punktraster des Sprite 2 übertragen.

SPRSAV A$,1

Dem Sprite 1 wird das Punktraster von der Variablen A$ übermittelt.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
› COMMODORE 128	:	SPRSAV Zahl,Variable SPRSAV Variable,Zahl
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SQ

Befehlsart: Funktion

SQ (**Zahl**)

Beschreibung: Anhand dieser Funktion kann man Informationen über den Zustand der Tonwarteschlangen abrufen.

Als erstes muß man wissen, daß die Tonerzeugung bei den Heimcomputern ein kniffliges Kapitel ist. Am besten läßt es sich durch viel Ausprobieren erforschen.

Durch die Zahl **Zahl** wird der Tonkanal festgelegt. Dabei ist der Tonkanal A die **Zahl** 1, B die **Zahl** 2 und C die **Zahl** 4 (im Handbuch für den Schneider CPC 464 ist an dieser Stelle ein Druckfehler).

Als Ergebnis erhält man ein BYTE. Je nach dem dort (in diesem BYTE) gesetzten BIT kann man die Information des Tonkanales entschlüsseln. Deshalb ist es günstiger, das Ergebnis der Funktion SQ in eine binäre Zahl (siehe BIN$) umzuwandeln.

Beispiel:

```
? BIN$(SQ(1),8)
ØØØØØ1ØØ
```

!!!!!LLL	BITØ-BIT2	geben Aufschluß über die freien
!!!!!		Warteplätze in der betreffenden Warte-
		schlange
!!!!L	BIT3	Warten auf Kanal A
!!!L	BIT4	Warten auf den Kanal B
!!L	BIT5	Warten auf den Kanal C
!L	BIT6	Haltepunkt am Beginn der Warteschlange
L	BIT7	wird gesetzt, wenn Kanal aktiv ist

In diesem Beispiel hat der Kanal A vier freie Warteplätze.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	SQ(Zahl)
› SCHNEIDER CPC664,CPC6128	:	SQ(Zahl)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SQR

Befehlsart: Funktion

SQR(**Zahl**)

Beschreibung: Die Funktion SQR (square root = Quadratwurzel) gibt die Quadratwurzel der **Zahl** wieder. Die **Zahl** muß dabei positiv sein, sonst erfolgt eine Fehlermeldung mit dem Wortlaut: "Improper Argument".

Beispiel:

```
10 CLS
20 INPUT "laenge des Quadrats";x
30 ORIGIN 0,0
40 DRAW 0,x
50 DRAW x,x
60 DRAW x,0
70 DRAW 0,0
80 PRINT:PRINT:PRINT
90 PRINT "der Inhalt des Quadrats betraegt ";SQR (x);" cm."
100 INPUT "noch eine Rechnung (j) ";r$:IF r$="j" THEN GOTO 10
```

Kompatibilitätsliste:

› ATARI 600,800,130XE	: SQR(Zahl)
› COMMODORE VC20,64	: SQR(Zahl)
› COMMODORE 16,116,PLUS4	: SQR(Zahl)
› COMMODORE 128	: SQR(Zahl)
› ENTERPRISE 64,128	: SQR(Zahl)
› MSX-STANDARD	: SQR(Zahl)
› SCHNEIDER CPC464	: SQR(Zahl)
› SCHNEIDER CPC664,CPC6128	: SQR(Zahl)
› SINCLAIR ZX81	: SQR Zahl
› SINCLAIR SPECTRUM	: SQR Zahl

SSHAPE

Befehlsart: Anweisung

SSHAPE **Variablenname,Zahl 1,Zahl 2,Zahl 3,Zahl 4**

Beschreibung: Einer Stringvariablen mit dem **Variablennamen** wird als Inhalt ein Bildschirmausschnitt des Grafikschirmes übertragen.

Mit dem Befehl GSHAPE (siehe dort) kann man den Inhalt dieser Variablen wieder auf dem Bildschirm abbilden.

Die Information für die Variable kann bis zu 255 Zeichen umfassen.

Zahl 1 und **Zahl 3** geben die Eckpunkte des Bildschirmausschnitts auf der y-Achse an, also in waagrechter Richtung. Der so definierte Bildschirmbereich wird als Inhalt der Variablen mit dem **Variablennamen** übergeben.

Beispiel:

```
10 GRAPHIC1,1
20 FOR I=1 TO 75
30 CIRCLE 1,100,100,I*4
40 NEXT I
50 X=INT(RND(1)*32)*10
60 Y=INT(RND(1)*20)*10
70 GSHAPE A$,X,Y,X+9,Y+9
80 X=INT(RND(1)*32)*10
90 Y=INT(RND(1)*20)*10
100 GSHAPE A$,X,Y,0
110 GOTO 50
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
› COMMODORE 16,116,PLUS4	:	SSHAPE Variable,Zahl1,Zahl2,Zahl3,Zahl4
› COMMODORE 128	:	SSHAPE Variable,Zahl1,Zahl2,Zahl3,Zahl4
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

START

Befehlsart: Kommando

START

Beschreibung: Ein Programm im Rechner-Speicher wird gestartet. Dabei wirkt der Befehl wie das Kommando RUN (siehe dort).

Ist im Speicher des Rechners kein Programm vorhanden, wird mit START ein Programm geladen. Hierbei wird vom Kanal 1Ø6 (in der Regel Datenrecorder) geladen und sogleich gestartet. In ähnlicher Weise wirkt der Befehl RUN **String** (siehe dort).

Beispiel:

START

Kompatibilitätsliste:

```
  ATARI 600,800,130XE        :  RUN "Programmname"
  COMMODORE VC20,64          :
  COMMODORE 16,116,PLUS4     :
  COMMODORE 128              :  RUN "Programmname",DZahl1,UZahl
> ENTERPRISE 64,128          :  START
  MSX-STANDARD               :
> SCHNEIDER CPC464           :  RUN ""
> SCHNEIDER CPC664,CPC6128   :  RUN ""
  SINCLAIR ZX81              :
  SINCLAIR SPECTRUM          :
```

STASH

Befehlsart: Kommando

STASH **Zahl 1,Zahl 2,Zahl 3,Zahl 4**

Beschreibung: Ein bestimmter Speicherbereich des BASIC-Arbeitsspeichers wird in eine bestimmte Arbeitsbank übertragen.

Diese Methode der Speicherung nennt man auch RAMDISK. Bei ihr wird ein Teil des Rechnerspeichers als Zwischenspeicher verwendet. Der Vorteil dieser Methode ist die schnellere Datenübertragung, verglichen mit Kassettenrecorder und Floppylaufwerk. Der Nachteil liegt in einem möglichen Speicherverlust, zum Beispiel durch Stromausfall.

Der Umkehrbefehl zu STASH ist FETCH (siehe dort).

Zahl 1 gibt die Anzahl der Bytes an, die in die Speicherbank übertragen werden sollen. Sie kann Werte zwischen 1 und 65535 annehmen.

Zahl 2 gibt die Nummer der Speicherzelle an, in der das erste Byte des Datenblocks steht. Auch ihr Wert kann zwischen 1 und 65535 liegen.

Zahl 3 bezeichnet die Nummer der Speicherbank, in die die Daten übertragen werden sollen.

Zahl 4 gibt die Nummer der Speicherzelle innerhalb der Datenbank an, nach der das erste Byte der Daten übertragen werden soll. Der Wert liegt zwischen Ø und 65535.

Beispiel:

STASH 1Ø24,2Ø48,7,4Ø96

Der Rechner überträgt 1Ø24 Bytes ab der Adresse 2Ø48 in die Speicherbank 7 ab der Adresse 4Ø96.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  STASH Zahl1,Zahl2,Zahl3,Zahl4
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

STATUS

Befehlsart: Anweisung

STATUS **#Zahl,Variable**

Beschreibung: Dem Anwender wird übermittelt, ob die Ein- oder Ausgabe erfolgreich abgeschlossen wurde.

Die **Zahl** gibt die Kanalnummer an. Der Kanal muß zunächst mit dem OPEN-Befehl (siehe dort) geöffnet werden. Diese **Zahl** hat einen ganzzahligen Wert zwischen 1 und 7.

Der **Variablen**name gibt eine numerische Variable an. Ihr wird der Wert der Statusabfrage zugewiesen. Ist er größer als 127, so ist ein Fehler aufgetreten, der sich mit Hilfe der Fehlermeldungstabelle "dekodieren" läßt.

Beispiele:

STATUS #1,I

STATUS #2,BA

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	STATUS #Zahl,Variable
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

STEP

Befehlsart: Anweisung

STEP **Zahl**

Beschreibung: Die **Zahl** gibt die Schrittweite beim Zählen einer FOR...TO-Schleife wieder (siehe FOR...TO...STEP... NEXT). Dieser Befehl wird auch teilweise bei anderen Befehlen verwendet (z.B. bei den MSX-Computern bei dem Befehl CIRCLE).

Kompatibilitätsliste:

› ATARI 600,800,130XE	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› COMMODORE VC20,64	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› COMMODORE 16,116,PLUS4	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› COMMODORE 128	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› ENTERPRISE 64,128	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› MSX-STANDARD	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› SCHNEIDER CPC464	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› SCHNEIDER CPC664,CPC6128	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› SINCLAIR ZX81	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
› SINCLAIR SPECTRUM	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3

STICK

Befehlsart: Funktion

STICK (**Zahl**)

Beschreibung: Mit Hilfe des Befehls STICK kann man die augenblickliche Stellung des Joysticks bestimmen.
Die **Zahl** gibt die Nummer des Joysticks wieder. Ihr Wert beträgt entweder Ø oder 1.
Der zurückgegebene Wert liegt zwischen 5 und 15.

Beispiel:

```
10 IF STICK(0)<>15 THEN PRINT "DER JOYSTICK WURDE BEWEGT."
20 GOTO 10
```

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  STICK(Zahl)
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :  JOY(Zahl)
  COMMODORE 128             :  JOY(Zahl)
  ENTERPRISE 64,128         :  JOY(Zahl)
  MSX-STANDARD              :  STICK(Zahl)
  SCHNEIDER CPC464          :  JOY(Zahl)
  SCHNEIDER CPC664,CPC6128  :  JOY(Zahl)
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

STICK

Befehlsart: Funktion

STICK (**Zahl**)

Beschreibung: Dieser Befehl entspricht dem STICK-Befehl im ATARI-BASIC (siehe dort). Die **Zahl** kann allerdings zwischen Ø und 2 liegen.

Es entspricht
Zahl
Ø Cursor-Tasten
1 Joystick 1
2 Joystick 2

Der zurückgegebene Wert liegt bei dieser Abfrage zwischen Ø und 8.

Beispiel:

```
1Ø  P=STICK(Ø)
2Ø  PRINT P
3Ø  GOTO 1Ø
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	STICK(Zahl)
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	JOY(Zahl)
COMMODORE 128	:	JOY(Zahl)
ENTERPRISE 64,128	:	JOY(Zahl)
> MSX-STANDARD	:	STICK(Zahl)
SCHNEIDER CPC464	:	JOY(Zahl)
SCHNEIDER CPC664,CPC6128	:	JOY(Zahl)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

STOP

Befehlsart: Kommando

STOP

Beschreibung: Ein Programm wird an einer bestimmten Stelle unterbrochen. Mit dem Befehl CONT (siehe dort) kann man das Programm wieder fortsetzen.

Das STOP-Kommando ist vor allem für die Fehlersuche in einem Programm geeignet. An den jeweils geeigneten Stellen wird das Programm unterbrochen. Danach hat man Gelegen-

heit, mit PRINT (siehe dort) die Variablen abzufragen und Veränderungen vorzunehmen, falls etwa die Variablen sinnlose Werte annehmen konnten, die die Programmwirkung beeinträchtigen.

Beispiel:

```
10 FOR i=-5 TO 5
20 IF i>-1 THEN STOP
30 x=12/i
40 PRINT x
50 NEXT i
```

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  STOP
> COMMODORE VC20,64         :  STOP
> COMMODORE 16,116,PLUS4    :  STOP
> COMMODORE 128             :  STOP
> ENTERPRISE 64,128         :  STOP
> MSX-STANDARD              :  STOP
> SCHNEIDER CPC464          :  STOP
> SCHNEIDER CPC664,CPC6128  :  STOP
> SINCLAIR ZX81             :  STOP
> SINCLAIR SPECTRUM         :  STOP
```

STOP ON/OFF

Befehlsart: Anweisung

STOP ON
STOP OFF
STOP STOP

Beschreibung: STOP ON bzw. STOP OFF unterstützt den ON STOP GOSUB-Befehl (siehe dort). Bei STOP ON wird die Unterbrechung gültig, bei STOP OFF ungültig. Mit STOP STOP wird das Programm zum Halten gebracht.

Beispiel:
```
10 ON STOP GOSUB 100
20 STOP ON
30 PRINT "BITTE ANHALTEN!"
40 GOTO 30
100 STOP OFF
110 PRINT "BEIM NAECHSTEN MAL GEHT ES!"
120 RETURN
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	STOP ON STOP OFF STOP STOP
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

STR$

Befehlsart: Funktion

STR$(**Zahl**)

Beschreibung: Dieser Befehl ist die Umkehrfunktion zu VAL (siehe dort). Die Funktion STR$ formt eine Zahl in eine Zeichenkette um.

Ist die umzuwandelnde **Zahl** positiv, dann ist das erste Zeichen der Zeichenkette eine Leerstelle. Diese Leerstelle bedeutet nichts anderes als das Plus-Zeichen.

Beispiele:
```
10 X=1200
20 A$=STR$(X)
30 PRINT A$,RIGHT$(A$,3)
RUN
 1200       200

READY.
```

```
10 X=12
20 PRINT STR$(X)
30 PRINT VAL(STR$(X))
RUN
 12
 12

READY.

10 X=-1
20 PRINT STR$(X)
RUN
-1

READY.
```

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  STR$(Zahl)
> COMMODORE VC20,64         :  STR$(Zahl)
> COMMODORE 16,116,PLUS4    :  STR$(Zahl)
> COMMODORE 128             :  STR$(Zahl)
> ENTERPRISE 64,128         :  STR$(Zahl)
> MSX-STANDARD              :  STR$(Zahl)
> SCHNEIDER CPC464          :  STR$(Zahl)
> SCHNEIDER CPC664,CPC6128  :  STR$(Zahl)
> SINCLAIR ZX81             :  STR$ Zahl
> SINCLAIR SPECTRUM         :  STR$ Zahl
```

STRIG

Befehlsart: Funktion

STRIG (**Zahl**)

Beschreibung: Der Befehl gleicht der STRIG-Anweisung anderer Rechner (siehe dort). Allerdings kann man durch Angabe der **Zahl** entscheiden, ob man die Leertaste (**Zahl**=Ø), Joystick 1 (**Zahl**=1) oder Joystick 2 (**Zahl**=4) abfragen will.
Als zurückgegebener Wert bedeutet

-1 Leertaste oder Joystick-Feuerknopf gedrückt.
Ø Leertaste oder Joystick-Feuerknopf nicht gedrückt.

Beispiel:

```
10 IF STRIG(0)=0 THEN GOTO 10 ELSE GOTO 20
20 PRINT "Leertaste gedrueckt"
```

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  STRIG(Zahl)
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  STRIG(Zahl)
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

STRIG ON/OFF

Befehlsart: Anweisung

STRIG (**Zahl**) ON
STRIG (**Zahl**) OFF

Beschreibung: Man kann die Anweisung einsetzen, falls bei einem ON STRIG GOSUB-Befehl (siehe dort) eine Unterbrechung erfolgen soll. Sie läßt sich durch Betätigung der Leertaste oder des Joystick-Feurknopfes auslösen.
Mit

STRIG (**Zahl**) ON

wird der Befehl gültig, und es kann ein Sprung zu der Zeile erfolgen, die durch GOSUB definiert wurde.
Mit

STRIG (**Zahl**) OFF

wird der Befehl ungültig.
Mit

STRIG (**Zahl**) STOP

wird das Programm angehalten.

Die **Zahl** gibt an, ob die Unterbrechung durch die Leertaste (**Zahl** = Ø) oder die Joysticks (**Zahl** = 1, 2, 3 oder 4) ausgelöst wird.

Beispiel:

```
10 ON STRIG GOSUB 100
20 STRIG(0) ON
30 PRINT "Bitte Leertaste druecken!"
40 GOTO 30
100 STRIG(0) OFF
110 PRINT "Das ging nur einmal!"
120 FOR N=1 TO 1000:NEXT
130 RETURN
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
› MSX-STANDARD              :  STRIG ON
                               STRIG OFF
                               STRIG STOP
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

STRING

Befehlsart: Anweisung

STRING **Variablenliste**

Beschreibung: Die Variablen in der **Variablenliste** werden als Stringvariablen festgelegt. Ähnlich dem Befehl DIM (siehe dort) lassen sich mit STRING auch Felder definieren.

Definition von Stringvariablen:

STRING **Variablenname**

legt mit dem **Variablennamen** eine Variable als Stringvariable fest.

Definition eines String-Feldes:

Die möglichen Formate für diesen Befehl sind

STRING **Variablenname(Zahl 2,Zahl 4...)**

oder

STRING **Variablenname(Zahl** 1 TO **Zahl 2,Zahl** 3 TO **Zahl 4...)#Zahl Ø**

Bei der Definition eines String-Feldes gibt der **Variablenname** die Namen der indizierten Variablen an. **Zahl 2, Zahl 4...** usw. kennzeichnen die Obergrenze eines Feldes. **Zahl 1, Zahl 3...** usw. definieren die Untergrenze eines Feldes. Normalerweise haben diese Zahlen den Wert Ø.
Mit **Zahl Ø** kann man die Länge des Strings vorgeben.

Anmerkung: Die Befehlsformate lassen sich auch kombinieren, wie aus dem Beispiel ersichtlich.

Beispiel:

```
STRING APFEL,BIRNE(-3TO12,13)#1Ø
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	STRING Variablenliste
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

STRING$

Befehlsart: Funktion

STRING$(**Zahl**,"**Textzeichen**")

Beschreibung: Es wird ein String erzeugt, der im Höchstfall 255 Zeichen lang sein kann. Dabei wird ein **Textzeichen** **Zahl**mal zu einem Leerstring gezählt.

Wenn man statt eines einzelnen **Textzeichen**s einen String einfügt, verwendet der STRING$-Befehl für die Stringverkettung nur das erste Zeichen.

Die **Zahl** darf Werte von Ø bis 255 annehmen. Beim Wert Ø wird ein Leerstring, beim Wert 255 ein 255 Zeichen langer String zurückgegeben.

Beispiel:

```
10 FOR i=0 TO 255
20 PRINT  STRING$(i,"#")
30 NEXT i
```

Anmerkung zur Kompatibilität: Der STRING$-Befehl kann bei Rechnern, die nicht über diese Funktion verfügen, wie folgt simuliert werden:

```
a$=STRING$(Zahl,Textzeichen)
```

kann durch

```
FOR i=1 TO Zahl:a$=a$+Textzeichen:NEXT i
```

ersetzt werden.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  STRING$(Zahl,"Textzeichen")
                               STRING$(Zahl,ASC-Zahl)
> SCHNEIDER CPC464          :  STRING$(Zahl,"Textzeichen")
> SCHNEIDER CPC664,CPC6128  :  STRING$(Zahl,"Textzeichen")
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

SWAP

Befehlsart: Anweisung

SWAP **Variable 1,Variable 2**

Beschreibung: Die SWAP-Anweisung vertauscht den Inhalt von **Variable 1** mit dem von **Variable 2.** Dabei müssen die beiden Variablen vom gleichen Typ sein.

Beispiel:

```
10 INPUT "Zahl A";A
20 INPUT "Zahl B";B
30 SWAP A,B
40 PRINT "Zahl A";A
50 PRINT "Zahl B";B
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  SWAP Variable1,Variable2
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

SWAP

Befehlsart: Anweisung

SWAP **Zahl 1,Zahl 2,Zahl 3,Zahl 4**

Beschreibung: Durch diesen Befehl lassen sich Daten des BASIC-Speichers mit denen der Speicherbank **Zahl 3** vertauschen. Die **Zahl 1** gibt die Anzahl der auszutauschenden Daten in BYTE an. Die **Zahl 2** gibt die Adresse im BASIC-Arbeitsspeicher an, ab der die Daten getauscht werden sollen, und die **Zahl 4** gibt die Adresse der Speicherbank an.

Beispiel:

SWAP 1Ø24,4Ø96,6,2Ø48

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
› COMMODORE 128	:	SWAP Zahl1,Zahl2,Zahl3,Zahl4
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SYMBOL

Befehlsart: Anweisung

SYMBOL **ASCII-Zeichen,Zahl 1,Zahl 2,Zahl 3,Zahl 4,Zahl 5, Zahl 6,Zahl 7,Zahl 8**

Beschreibung: Durch den SYMBOL-Befehl lassen sich eigene Grafikzeichen erstellen, denn die vom Rechner angebotenen Zeichen reichen nicht für jeden Fall aus.

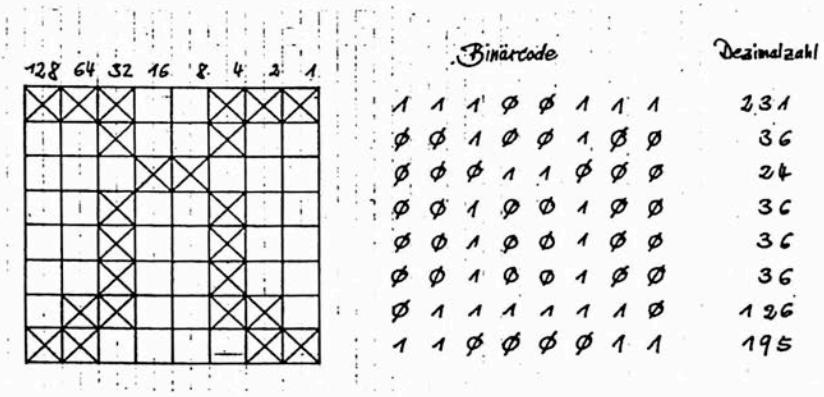
Der Wert des **ASCII-Zeichens** liegt zwischen Ø und 255. Das bedeutet: Eines der vom Rechner angebotenen Zeichen wird mit dem selbst definierten Zeichen überschrieben und steht dann nicht mehr zur Verfügung.

Ist die gewünschte ASCII-Zahl allerdings kleiner als 24Ø, so muß man den Computer erst anhand des Befehls SYMBOL AFTER (siehe dort) vorbereiten.

Zahl 1 bis **Zahl 8** ergeben sich aus der 8x8-Matrix, in der das neue Zeichen entsteht (siehe Beispiel).

Beispiel:

```
10 CLS
20 SYMBOL 255,231,36,24,36,36,36,126,195
30 PRINT CHR$(255)
40 PRINT
```



Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :
      SYMBOL Zahl,Zahl1,Zahl2,Zahl3,Zahl4,Zahl5,Zahl6,Zahl7,Zahl8
> SCHNEIDER CPC664,CPC6128  :
      SYMBOL Zahl,Zahl1,Zahl2,Zahl3,Zahl4,Zahl5,Zahl6,Zahl7,Zahl8
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

SYMBOL AFTER

Befehlsart: Anweisung

SYMBOL AFTER **ASCII-Code**

Beschreibung: Mit dem Befehl SYMBOL AFTER kann der Benutzer jenen Bereich in der ASCII-Tabelle festlegen, in dem er eigene Zeichen definieren möchte. Der Computer selbst bietet ihm die Werte zwischen 24Ø und 255 an. Versucht man nun ein mit SYMBOL (siehe dort) definiertes Zeichen zum Beispiel auf den ASCII-Wert 9Ø zu legen, so erfolgt eine Fehlermeldung. Erst durch

SYMBOL AFTER 9Ø

wird jener Bereich der ASCII-Tabelle, in der eigene Zeichen abgelegt werden können, auf die Werte 9Ø bis 255 erweitert. Und nun kann man das Zeichen anhand des SYMBOL-Befehls auch auf den ASCII-Wert 9Ø ablegen.

Beispiel:

```
90 CLS
100 SYMBOL AFTER 90
110 SYMBOL 90,231,36,24,36,36,126,195
120 PRINT CHR$(90):PRINT:PRINT:PRINT CHR$(90)
130 GOTO 130
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	SYMBOL AFTER Zahl
› SCHNEIDER CPC664,CPC6128	:	SYMBOL AFTER Zahl
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

SYS

Befehlsart: Anweisung

SYS **Zahl,Parameterliste**

Beschreibung: Mit diesem Befehl kann der erfahrene Programmierer ein Maschinenprogramm vom BASIC aus aufrufen.

Die **Zahl** gibt die Nummer der Speicherzelle an, in der das Maschinenprogramm beginnt. Sie kann zwischen Ø und 65535 liegen.

Mit Hilfe der **Parameterliste** kann der Anwender Variablen in das Maschinenprogramm übertragen. Auf die **Parameterliste** kann man gegebenenfalls verzichten.

Beispiele:

SYS 2Ø48

SYS 4Ø96 12,X,Y

Anmerkung: Das Maschinenprogramm muß mit der Anweisung RTS (Return) abgeschlossen werden, damit zum BASIC zurückgekehrt wird.

Dieser Befehl sollte nur verwendet werden, falls man über Erfahrungen im Programmieren in Maschinensprache ver-

fügt. Durch fehlerhafte Eingabe eines SYS-Befehls kann der Computer sich "aufhängen", das heißt, er ist in einem Blockierzustand, aus dem man nur durch ein RESET wieder herauskommt oder indem man den Rechner aus- und wieder einschaltet.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
> COMMODORE VC20,64         :  SYS Zahl,Parameterliste
> COMMODORE 16,116,PLUS4    :  SYS Zahl,Parameterliste
> COMMODORE 128             :  SYS Zahl,Parameterliste
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

TAB

Befehlsart: Funktion

TAB(**Zahl**)

Beschreibung: Der Textcursor wird auf die **Zahl**te Spalte einer Zeile gesetzt. Von dort ab wird der Text ausgegeben.
Es sind nur **Zahl**en von 1 bis 8Ø sinnvoll. Sie sind abhängig vom gewählten Bildschirm-Modus, sofern sich dieser verändern läßt.

Beispiel:

```
10 FOR I=1 TO 80
20 PRINT TAB(I)"#"
30 NEXT I
40 FOR I=80 TO 1 STEP -1
50 PRINT TAB (I)"*"
60 NEXT I
70 GOTO 10
```

Anmerkung: Die Funktion TAB ist nur sinnvoll in Verbindung mit einer PRINT-Anweisung.

Anmerkung zur Kompatibilität: Bei den Sinclair-Computern hat die Funktion das Format TAB **Zahl**.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
> COMMODORE VC20,64         :  TAB(Zahl)
> COMMODORE 16,116,PLUS4    :  TAB(Zahl)
> COMMODORE 128             :  TAB(Zahl)
> ENTERPRISE 64,128         :  TAB(Zahl)
> MSX-STANDARD              :  TAB(Zahl)
> SCHNEIDER CPC464          :  TAB(Zahl)
> SCHNEIDER CPC664,CPC6128  :  TAB(Zahl)
> SINCLAIR ZX81             :  TAB Zahl
> SINCLAIR SPECTRUM         :  TAB Zahl
```

TAG

Befehlsart: Anweisung

TAG #**Kanalnummer**

Beschreibung: Die folgende PRINT-Anweisung (siehe dort) wird nicht an der Textcursor-Stellung, sondern an der Grafikcursor-Stellung ausgeführt. Dabei wird die linke obere Ecke des ersten auszugebenden Zeichens an die Grafikcursor-Position gesetzt.

Zu beachten ist dabei, daß hinter jeder PRINT-Anweisung ein Strichpunkt (;) geschrieben wird, da sonst der Computer hinter den Textausdruck die Grafiksymbole für die Steuerzeichen (z. B. Line Feed) anhängt.

Wird keine **Kanalnummer** angegeben, wählt der Rechner Kanal #Ø, also den Bildschirm, an.

Der TAG-Befehl wird mit dem Befehl TAGOFF (siehe dort) aufgehoben.

Beispiel:

```
10 MODE 2
20 TAG
30 PLOT 0,16:DRAW 200,16:PRINT "NEUE ";
40 PRINT "POSITION";
50 TAGOFF
60 PRINT "ALTE POSITION";
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :  TAG #Kanalnummer
> SCHNEIDER CPC664,CPC6128  :  TAG #Kanalnummer
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

TAGOFF

Befehlsart: Anweisung

TAGOFF **Kanalnummer**

Beschreibung: Die mit dem TAG-Befehl (siehe dort) erreichte Textausgabe an jener Stelle, an der sich der Grafikcursor befindet, wird mit dem Befehl TAGOFF wieder aufgehoben. Alle nach der TAGOFF-Anweisung auftretenden PRINT-Anweisungen (siehe dort) werden wieder an der Textcursor-Position ausgeführt.

Gibt man keine **Kanalnummer** an, wählt der Rechner Kanal Ø, also den Bildschirm, an.

Beispiel:

```
10 MODE 2
20 TAG
30 PLOT 0,16:DRAW 200,16:PRINT "NEUE ";
40 PRINT "POSITION"
50 TAGOFF
60 PRINT "TEXT"
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :  TAGOFF #Kanalnummer
> SCHNEIDER CPC664,CPC6128  :  TAGOFF #Kanalnummer
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

TAN

Befehlsart: Funktion

TAN(**Zahl**)

Beschreibung: Die Funktion gibt den Tangens der **Zahl** zurück. Im normalen Fall geschieht dies im Bogenmaß, bei einigen Rechnern allerdings auch im Winkelmaß (siehe auch SIN, COS, ASN usw.).

Beispiele:

```
?TAN (Ø)
 Ø
READY
```

```
10 FOR I=1 TO 5
20 PRINT TAN (I)
30 NEXT I
```

Anmerkung zur Kompatibilität: Bei den Sinclair-Computern hat der Befehl das Format TAN **Zahl.**

Achtung: Manche Rechner können bei ungeraden Vielfachen von PI/2 (= (2k+1) PI/2 "aussteigen", da der Tangens dann gegen Unendlich geht. Andere Rechner bringen statt dessen Fehlermeldungen.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
› COMMODORE VC20,64	:	TAN(Zahl)
› COMMODORE 16,116,PLUS4	:	TAN(Zahl)
› COMMODORE 128	:	TAN(Zahl)
› ENTERPRISE 64,128	:	TAN(Zahl)
› MSX-STANDARD	:	TAN(Zahl)
› SCHNEIDER CPC464	:	TAN(Zahl)
› SCHNEIDER CPC664,CPC6128	:	TAN(Zahl)
› SINCLAIR ZX81	:	TAN Zahl
› SINCLAIR SPECTRUM	:	TAN Zahl

TANH

Befehlsart: Funktion

TANH(**Zahl**)

Beschreibung: Mit dieser Funktion erhält der Anwender den TANGENSHYPERBOLICUS der **Zahl.**

Beispiele:

```
PRINT TANH(1)
```

```
A=TANH(B)
```

Anmerkung: Die Funktion ist nur für Anwender von Interesse, die mit den Winkelfunktionen vertraut sind.

Anmerkung zur Kompatibilität: Bei Rechnern, deren BASIC nicht über diesen Befehl verfügt, kann er wie folgt simuliert werden:

```
TANH(Zahl)
```

entspricht dann

```
(EXP(-Zahl)/(EXP(Zahl)+EXP(-Zahl))*2+1
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	TANH(Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

TEMPO

Befehlsart: Anweisung

TEMPO **Zahl**

Beschreibung: Durch diesen Befehl kann man das Spieltempo für die Musiknoten festlegen, die mit Hilfe der PLAY-Anweisung (siehe dort) ausgegeben werden.

Mit der **Zahl** läßt sich die relative Dauer bestimmen, mit der die Musiknoten gespielt werden. Sie kann ganzzahlige Werte zwischen Ø und 255 annehmen.

Ist der Wert Ø, wird die Note als Dauerton gespielt.

Sonst berechnet sich die Tondauer nach der Formel

Tondauer = 19.22/**Zahl** Sekunden

Beispiel:

TEMPO 8

Eine Note wird 2.4 Sekunden lang gespielt.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
› COMMODORE 128	:	TEMPO Zahl
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

TEST

Befehlsart: Funktion

TEST(**Zahl 1,Zahl 2**)

Beschreibung: Mit der Funktion kann abgefragt werden, welcher Farbstift an dem durch **Zahl 1** und **Zahl 2** angegebenen Pixel (= Bildschirmpunkt) wirkt. **Zahl 1** bezeichnet die absolute x-Koordinate, **Zahl 2** die absolute y-Koordinate.

Beispiel:

```
10 CLS
20 INK 2,2
30 PAPER 2
40 PRINT"                                    "
50 PAPER 0
60 PRINT "FARBSTIFT" TEST(100,100)"am Punkt(100,100)"
70 PRINT "FARBSTIFT" TEST(100,399)"am Punkt(100,399)"
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	TEST(Zahl1,Zahl2)
› SCHNEIDER CPC664,CPC6128	:	TEST(Zahl1,Zahl2)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

TESTR

Befehlsart: Funktion

TESTR(**Zahl 1,Zahl 2**)

Beschreibung: Mit der Funktion TESTR kann man abfragen, welcher Farbstift an der Pixel-Stellung wirkt, die gegenüber der augenblicklichen Cursorstellung um **Zahl 1** (= x-Koordinate) und **Zahl 2** (= y-Koordinate) verschoben wurde.

Beispiel:

```
10 CLS
20 ORIGIN 100,100
30 DRAW 200,100
40 PRINT TESTR (-100,-50)
50 PRINT TESTR (4,4)
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	TESTR(Zahl1,Zahl2)
› SCHNEIDER CPC664,CPC6128	:	TESTR(Zahl1,Zahl2)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

TEXT

Befehlsart: Kommando

TEXT **Zahl**

Beschreibung: Mit diesem Kommando schaltet man entweder vom Grafik- auf den Textbetrieb um, von 40-Zeichen- auf 80-Zeichen-Darstellung, oder umgekehrt.

Geben wir TEXT ohne **Zahl** ein, wird der Grafikbetrieb ausgeschaltet und auf den zuvor verwendeten Textbetrieb zurückgeschaltet. Ansonsten gibt man mit der **Zahl**, die entweder 40 oder 80 betragen kann, die Zeilenbreite an.

Beispiel:

```
100 PRINT "DIES IST DIE UEBERSCHRIFT"
110 TEXT 80
120 PRINT "UND DIES IST DER TEXT, DER AUF DIE UEBERSCHRIFT FOLGT"
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	GRAPHICS Zahl
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	GRAPHIC Zahl
COMMODORE 128	:	GRAPHIC Zahl
› ENTERPRISE 64,128	:	TEXT Zahl
MSX-STANDARD	:	SCREEN Zahl
SCHNEIDER CPC464	:	MODE Zahl
SCHNEIDER CPC664,CPC6128	:	MODE Zahl
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

THEN

Befehlsart: Anweisung

IF**Bedingung**...THEN**Anweisung**

Beschreibung: Die Anweisung THEN ist Bestandteil des IF... THEN-Befehls (siehe dort). Er bewirkt, daß eine **Anweisung** oder auch ein Befehl ausgeführt wird, wenn eine **Bedingung** erfüllt wird.

Beispiel:

```
10 PRINT "A= ";
20 INPUT A
30 PRINT "In welchen Schritten soll ich die Vari- able >A< zaehlen?"
40 INPUT B
50 K=1
60 C=A+B*K:PRINT C
70 PRINT "WEITERZAEHLEN?"
80 A$=INKEY$:IF A$=""THEN GOTO 80
90 IF A$="J" OR A$="j" THEN K=K+1: GOTO 60
100 CLEAR:GOTO 10
```

Kompatibilitätsliste:

```
> ATARI 600,800,130XE       :  IF Bedingung THEN Anweisung
> COMMODORE VC20,64         :  IF Bedingung THEN Anweisung
> COMMODORE 16,116,PLUS4    :  IF Bedingung THEN Anweisung ELSE Anweisung
> COMMODORE 128             :  IF Bedingung THEN Anweisung ELSE Anweisung
> ENTERPRISE 64,128         :  IF Bedingung THEN Anweisung ELSE Anweisung
> MSX-STANDARD              :  IF Bedingung THEN Anweisung ELSE Anweisung
> SCHNEIDER CPC464          :  IF Bedingung THEN Anweisung ELSE Anweisung
> SCHNEIDER CPC664,CPC6128  :  IF Bedingung THEN Anweisung ELSE Anweisung
> SINCLAIR ZX81             :  IF Bedingung THEN Anweisung
> SINCLAIR SPECTRUM         :  IF Bedingung THEN Anweisung
```

TIME

Befehlsart: Anweisung

TIME **String**

Beschreibung: Dem Rechner wird die augenblickliche Zeit zugewiesen. Diese ist im **String** niedergelegt. Er ist acht Zeichen lang und hat das Format

"HH:MM:SS"

wobei HH für die Stunden, MM für die Minuten und SS für die Sekunden steht. Bei einstelligen Eingaben muß man also jeweils eine Ø mit eingeben.

Beispiel:

Für den Zeitpunkt 13 Uhr 1 Minute und Ø Sekunden wird eingegeben

TIME "13:Ø1:ØØ"

Anmerkung zur Kompatibilität: Bei anderen Rechner-Typen ist diese Anweisung als TIME$ definiert.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
> COMMODORE VC20,64	:	TI-Zahl
> COMMODORE 16,116,PLUS4	:	TI-Zahl
> COMMODORE 128	:	TI-Zahl
> ENTERPRISE 64,128	:	TIME-String
> MSX-STANDARD	:	TIME-Zahl
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

TIME

Befehlsart: Funktion

TIME

Beschreibung: Mit dieser Funktion kann man die Zeit abfragen, die seit dem Einschalten des Computers oder seit einem RESET des Rechners vergangen ist. Die interne Uhr wird bei der Datenübertragung auf die Datenkassette gestoppt. Die Zeit wird dabei in 1/3ØØ Sekunden-Schritten gezählt.

Beispiel:

```
10 PRINT INT(TIME/300)
20 FOR t=1 TO 770:NEXT
30 CLS
40 GOTO 10
```

Dieses Programm zählt links oben am Bildschirm im Sekundentakt.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
› COMMODORE VC20,64         :  TI
› COMMODORE 16,116,PLUS4    :  TI
› COMMODORE 128             :  TI
  ENTERPRISE 64,128         :
› MSX-STANDARD              :  TIME
› SCHNEIDER CPC464          :  TIME
› SCHNEIDER CPC664,CPC6128  :  TIME
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

TIME$

Befehlsart: Funktion/Anweisung

TIME$
TIME$=**String**

Beschreibung: Mit dem Befehl TIME$ kann man die rechnerinterne Uhr abfragen bzw. neu einstellen.
Das Format

TIME$

dient zur Abfrage der Zeit.
Der **String** wird benötigt, um die Uhr neu einzustellen. Er hat die Form
HHMMSS
und besteht also aus insgesamt 6 Zeichen.

HH gibt die Stunden an (ØØ bis 23)
MM gibt die Minuten an (ØØ bis 59)
SS gibt die Sekunden an (ØØ bis 59).

Beispiele:

```
10 INPUT "ZEIT (HHMMSS)",TI$
20 PRINT CHR$(19)"AUGENBLICKLICHE ZEIT: "LEFT$(TI$,2)":"MID$(TI$,3,2)":"MID$(TI$
,5)
30 GOTO 20
```

(Für Commodore 16, VC2Ø, 64, 116, 128, plus4.)

```
10 PRINT CHR$(26);TIME $
20 GOTO 10
```

(Für Enterprise 64, 128.)

Beide Programme blenden oben links im Bildschirm den augenblicklichen Zeitstand ein.

Anmerkung zur Kompatibilität: Bei den Commodore-Rechnern kann man auch den Befehl TI$ (siehe dort) verwenden, da diese Computer nur 2-Zeichen-Variable verarbeiten und alle weiteren Zeichen einer Variablen ignorieren.

Kompatibilitätsliste:

ATARI 600,800,130XE		
› COMMODORE VC20,64	TI$	TI$=String
› COMMODORE 16,116,PLUS4	TI$	TI$=String
› COMMODORE 128	TI$	TI$=String
› ENTERPRISE 64,128	TIME$	TIME=String
MSX-STANDARD		
SCHNEIDER CPC464		
SCHNEIDER CPC664,CPC6128		
SINCLAIR ZX81		
SINCLAIR SPECTRUM		

TO

Befehlsart: Anweisung

FOR...TO...STEP

Beschreibung: TO ist ein Element des Befehls FOR...TO... STEP (siehe dort). Es bezeichnet im Format des Befehls

FOR **Variablenname=Zahl 1** TO **Zahl 2**

den Endwert einer Zählschleife.

Beispiel:

```
1 REM *H*E*O*L*P*E*
2 REM  *I*R*G*Y*H*N
5 PRINT "␣"
10 FOR X=150 TO 250
20 PRINT CHR$(X),
30 NEXT X
```

Kompatibilitätsliste:

> ATARI 600,800,130XE	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
> COMMODORE VC20,64	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
> COMMODORE 16,116,PLUS4	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
> COMMODORE 128	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
> ENTERPRISE 64,128	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
> MSX-STANDARD	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
> SCHNEIDER CPC464	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
> SCHNEIDER CPC664,CPC6128	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
> SINCLAIR ZX81	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3
> SINCLAIR SPECTRUM	:	FOR Variablenname=Zahl1 TO Zahl2 STEP Zahl3

TOGGLE

Befehlsart: Kommando

TOGGLE **Option**

Beschreibung: Mit diesem Befehl kann man bestimmte Rechnerattribute umschalten, die normalerweise nur eingeschaltet bzw. ausgeschaltet werden können. Die Attribute, die umgeschaltet werden sollen, sind mit der **Option** angegeben. Optionen sind:

FASTSAVE
INTERRUPT KEY
INTERRUPT NET
INTERRUPT STOP
KEY CLICK
REM 1
REM 2
SPEAKER
STATUS
TAPE SOUND

Die genaue Verwendung dieser Optionen ist dem Handbuch des ENTERPRISE zu entnehmen. Diesen Befehl sollten nur erfahrene Programmierer verwenden.

Beispiel:

TOGGLE KEY CLICK

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  TOGGLE OPTION
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

TRACE ON/OFF

Befehlsart: Kommando

TRACE ON TO **Kanalnummer**
TRACE ON
TRACE OFF

Beschreibung: Den Modus der Programmverfolgung schaltet man ein (TRACE ON) oder aus (TRACE OFF). (Siehe dazu auch TRON, TROFF.) Bei der Programmverfolgung wird immer die Nummer der Programmzeile angezeigt, in der sich der Rechner bei der Bearbeitung eines Programmes augenblicklich befindet.

Wenn man die Programmverfolgung auf ein anderes Gerät als den Bildschirm ausgeben will (zum Beispiel den Drukker), verwendet man das Format

TRACE ON TO **Kanalnummer**

Der mit der **Kanalnummer** bezeichnete Ausgabekanal kann Werten zwischen Ø und 255 entsprechen.

Beispiel:

```
11Ø TRACE ON
12Ø FOR I=1 TO 1ØØ
13Ø READ I
14Ø NEXT I
15Ø TRACE OFF
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:		
COMMODORE VC20,64	:		
COMMODORE 16,116,PLUS4	:	TRON	TROFF
COMMODORE 128	:	TRON	TROFF
› ENTERPRISE 64,128	:	TRACE ON	TRACE OFF
MSX-STANDARD	:	TRON	TROFF
SCHNEIDER CPC464	:	TRON	TROFF
SCHNEIDER CPC664,CPC6128	:	TRON	TROFF
SINCLAIR ZX81	:		
SINCLAIR SPECTRUM	:		

TRAP

Befehlsart: Anweisung

TRAP **Zeilennummer**

Beschreibung: Der TRAP-Befehl des Atari hat eine ähnliche Funktionsweise wie der Befehl ON ERROR GOTO (siehe dort) anderer Systeme.

Wurde der TRAP-Befehl ausgeführt, erfolgt beim nächsten auftretenden Fehler ein Sprung zu jener Programmzeile, die durch die **Zeilennummer** angegeben worden ist.

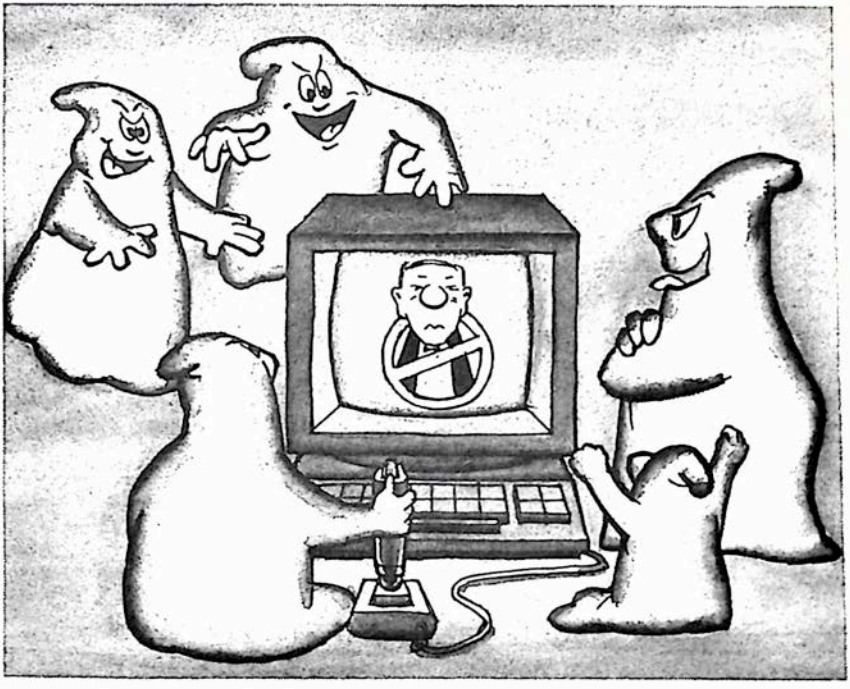
Soll die Fehlerbehandlung in dieser Art bestehen bleiben, muß das Programm erneut einen TRAP-Befehl durchlaufen.

Beispiel:

```
10 TRAP 100
20 PRINT "ZAHL  ;:INPUT Z
30 PRINT "10/ZAHL = ";10/Z
40 GOTO 20
100 PRINT "TEILUNG DURCH NULL!!":GOTO 10
```

Kompatibilitätsliste:

	ATARI 600,800,130XE :	
	COMMODORE VC20,64 :	
›	COMMODORE 16,116,PLUS4 :	TRAP Zeilennummer
›	COMMODORE 128 :	TRAP Zeilennummer
	ENTERPRISE 64,128 :	WHEN EXCEPTION USE Handlername
	MSX-STANDARD :	ON ERROR GOTO Zeilennummer
	SCHNEIDER CPC464 :	ON ERROR GOTO Zeilennummer
	SCHNEIDER CPC664,CPC6128 :	ON ERROR GOTO Zeilennummer
	SINCLAIR ZX81 :	
	SINCLAIR SPECTRUM :	



TROFF

Befehlsart: Kommando

TROFF

Beschreibung: Der Befehl zur Verfolgung des Programms TRON (siehe dort) wird abgeschaltet.

Beispiel:

```
 10 TRON
 20 PRINT "programm"
 30 PRINT "zum vergnuegen"
 40 GOTO 100
100 PRINT "^"
160 TROFF
170 PRINT "ENDE"
180 END
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  TROFF
> COMMODORE 128             :  TROFF
  ENTERPRISE 64,128         :  TRACE OFF
> MSX-STANDARD              :  TROFF
> SCHNEIDER CPC464          :  TROFF
> SCHNEIDER CPC664,CPC6128  :  TROFF
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

TRON

Befehlsart: Kommando

TRON

Beschreibung: Mit diesem Befehl erreicht man, daß das Programm verfolgt wird. Auf dem Bildschirm erscheint die Zei-

lennummer, in der sich das Programm gerade befindet. Das TRON-Kommando erleichtert somit das Auffinden von Fehlern.

Der Befehl TRON wird mit dem Kommando TROFF (siehe dort) aufgehoben.

Beispiel:

```
10 TRON:K=1
20 CLS:PRINT "Ich mache 10 Rechnungen."
30 INPUT "Welche Zahlen soll ich malnehmen";a,b
40 PRINT a;" * ";b;" = ";a*b
50 PRINT "Ich habe bisher ";k;" mal gerechnet."
60 PRINT "Ich rechne noch ";20-k;" mal."
70 k=k+1:IF k>5 THEN TROFF:IF k=21 THEN PRINT "Das war's! bis zum naechsten mal!
":END
80 INPUT "Welche neue Zahl soll ich mal ";a;" nehmen";c
```

Das Kommando TRON hilft hier, den Fehler im Programm zu entdecken und ihn in der richtigen Zeile (8Ø) ausfindig zu machen.

Anmerkung: Das TRON-Kommando läßt sich sowohl im Direktmodus als auch innerhalb eines Programms anwenden.

Anmerkung zur Kompatibilität: Beim Schneider CPC wird beim Einschalten der Programmverfolgung in der Programmausführung die Zeilennummer in eckigen Klammern angezeigt, bevor die entsprechende Zeile ausgeführt wird. Mögliche Bildschirmbilder werden dabei zerstört.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  TRON
> COMMODORE 128             :  TRON
> ENTERPRISE 64,128         :  TRACE ON
> MSX-STANDARD              :  TRON
> SCHNEIDER CPC464          :  TRON
> SCHNEIDER CPC664,CPC6128  :  TRON
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

TRUNCATE

Befehlsart: Funktion

TRUNCATE(**Zahl 1,Zahl 2**)

Beschreibung: Mit dieser Funktion kann man die Anzahl der Nachkommastellen einer Zahl verringern, die im Format des Befehls in **Zahl 1** niedergelegt ist.
Zahl 2 gibt an, wie viele Nachkommastellen **Zahl 1** besitzen soll.

Beispiel:

```
PRINT TRUNCATE(1.2343,2)
 1.23
READY
```

Anmerkung zur Kompatibilität: Bei Rechnern, deren BASIC nicht über diesen Befehl verfügt, läßt sich die Funktion wie folgt simulieren:

TRUNCATE(**Zahl 1,Zahl 2**)

entspricht

```
INT(ZAHL1*10↑ZAHL2)/10↑ZAHL2
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	TRUNCATE(Zahl1,Zahl2)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

TYPE

Befehlsart: Kommando

TYPE

Beschreibung: Mit diesem Befehl wird das BASIC verlassen, und man gelangt nach erneutem Drücken der ENTER-Taste in den Modus der Textverarbeitung. Drückt man statt ENTER eine andere Taste, wird das Kommando ignoriert.

Beispiel:

```
1110 INPUT "BEREIT ZUM SCHREIBEN";A$
1120 IF A$="JA" THEN TYPE
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
> ENTERPRISE 64,128         :  TYPE
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

UBOUND

Befehlsart: Funktion

UBOUND(**Variablenname**)
UBOUND(**Variablenname,Zahl**)

Beschreibung: Die Funktion gibt die obere Grenze eines durch DIM, NUMERIC oder STRING (siehe jeweils dort) definierten Feldes an.

Der **Variablenname** bezeichnet die Variable, die abgefragt werden soll.
Mit

UBOUND(**Variablenname**)

wird der größtmögliche Index bei einem eindimensionalen Feld abgefragt.
Mit

UBOUND(**Variablenname,Zahl**)

wird der größtmögliche Index bei einem Feld abgefragt, dessen Dimension die **Zahl** angibt. Ein dreidimensionales Feld hat drei Dimensionen; die **Zahl** kann somit die Werte 1, 2 oder 3 annehmen.

Beispiel:

```
PRINT UBOUND(A$)
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
> ENTERPRISE 64,128	:	UBOUND(Variable) UBOUND(Variable,Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

UCASE$

Befehlsart: Funktion

UCASE$(**String**)

Beschreibung: Mit dieser Funktion werden alle Buchstaben innerhalb eines **String**s in Großbuchstaben umgewandelt.
Die Funktion entspricht dem Befehl UPPER$ (siehe dort) bei anderen Rechner-Typen.
Der Umkehrbefehl lautet LCASE$ (siehe dort).

Beispiel:

```
PRINT UCASE$("Schneider Buch")
SCHNEIDER BUCH
READY
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	UCASE$(String)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	UPPER$(String)
SCHNEIDER CPC664,CPC6128	:	UPPER$(String)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

UNT

Befehlsart: Funktion

UNT(**Adresse**)

Beschreibung: Die Funktion UNT gibt den ganzzahligen Wert des angegebenen 16-Bit-**Adresse**n-Ausdrucks in der Klammer

zurück. Der Wert für die **Adresse** liegt zwischen Ø und 65535. Der zurückgegebene Wert kann zwischen -32768 und 32767 liegen.

Beispiele:

```
PRINT UNT (1)
11
READY

PRINT UNT (65535)
-1
READY

PRINT UNT (5ØØØØ)
-15536
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	UNT(Adresse)
› SCHNEIDER CPC664,CPC6128	:	UNT(Adresse)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

UNTIL

Befehlsart: Anweisung

DO LOOP... UNTIL

Beschreibung: Durch den UNTIL-Befehl wird eine Bedingung für das Ende einer mit DO LOOP (siehe dort) geschaffenen Schleife angegeben.

UPPER$

Befehlsart: Funktion

UPPER$("Textausdruck")
UPPER$(String)

Beschreibung: UPPER$ gibt den **Textausdruck** in reiner Großbuchstabenschrift wieder, falls dieser in Kleinbuchstabenschreibung oder in gemischter Schreibweise eingegeben worden ist.

Mit dieser Funktion oder der Umkehrfunktion LOWER$ (siehe dort) lassen sich Eingaben auswerten, die in unterschiedlichen Schreibweisen eingegeben wurden.

Beispiele:

```
10 INPUT a$
20 PRINT UPPER$(a$)
Schneider Computer Buecher

10 PRINT UPPER$("Schneider Computer Buecher")

SCHNEIDER COMPUTER BUECHER
```

Anmerkung: Diese Funktion betrifft ausschließlich Buchstaben. Satz- und Sonderzeichen bleiben davon unberührt.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	UCASE$(String)
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	UPPER$(String)
› SCHNEIDER CPC664,CPC6128	:	UPPER$(String)
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

USING

Befehlsart: Anweisung

PRINT USING

Beschreibung: Zusammen mit PRINT als PRINT USING (siehe dort) erlaubt die Anweisung die formatierte Ausgabe von Zahlen.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  PRINT #Kanal USING String;Ausgabeliste
> COMMODORE 128             :  PRINT #Kanal USING String;Ausgabeliste
> ENTERPRISE 64,128         :  PRINT #Kanal USING String;Ausgabeliste
> MSX-STANDARD              :  PRINT USING String;Ausgabeliste
> SCHNEIDER CPC464          :  PRINT #Kanal;USING String;Ausgabeliste
> SCHNEIDER CPC664,CPC6128  :  PRINT #Kanal;USING String;Ausgabeliste
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

USR

Befehlsart: Funktion

USR(**Adresse,Zahl**)

Beschreibung: Mit diesem Befehl kann man ein kleines Maschinenprogramm von BASIC aus aufrufen und dann den dort errechneten Wert einer Zahlenvariablen zuweisen.

Mit der **Adresse** wird die Startadresse des Maschinenprogramms angegeben. Ihr Wert liegt zwischen Ø und 65535.

Die **Zahl** ist eine Zahlenvariable. Sie belegt ein Doppelbyte. Das heißt, sie liegt im Wertebereich von -32768 bis 65535. Diese Zahl wird dem Maschinenprogramm zur Bearbeitung übergeben.

Anmerkung: Die Funktion sollte man nur verwenden, falls man mit dem Programmieren in Maschinensprache Erfahrung hat.

Anmerkungen zur Kompatibilität: Bei den ATARI-Rechnern kann statt der **Zahl** die Adresse der Variablen übergeben werden (siehe auch unter ADR).

Bei den Commodore-Rechnern und bei MSX-Computern hat der Befehl das Format

USR(**Zahl**)

Bei den Commodore-Rechnern muß die Adresse des Maschinenprogrammes in bestimmte Speicherzellen ge"poke"t werden. Bei den MSX-Rechnern wird dies mit Hilfe des DEFUSR-Befehls (siehe dort) bewerkstelligt.

Bei den SINCLAIR-Rechnern hat der Befehl das Format

USR(**Adresse**)

Es kann also kein Zahlenwert übermittelt werden.

Beispiele:

ATARI:

USR(1ØØØØ,ADR(a$))
USR(12ØØØ,123)

COMMODORE:

USR(A)

ENTERPRISE:

USR(1ØØØØ,123)

MSX:

USR(255)

SINCLAIR:

USR(5ØØØ)

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  USR(Adresse,Zahl)
› COMMODORE VC20,64         :  USR(Zahl)
› COMMODORE 16,116,PLUS4    :  USR(Zahl)
› COMMODORE 128             :  USR(Zahl)
› ENTERPRISE 64,128         :  USR(Adresse,Zahl)
› MSX-STANDARD              :  USRZahl1(Zahl)
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
› SINCLAIR ZX81             :  USR(Adresse)
› SINCLAIR SPECTRUM         :  USR(Adresse)
```

VAL

Befehlsart: Funktion

VAL(**String**)

Beschreibung: Aus einer Zeichenkette, die aus Ziffern besteht, macht die VAL-Funktion eine Zahl, mit der man rechnen kann. Dabei werden die Zeichen von links nach rechts geprüft und als Zahlen wiedergegeben, bis ein nicht-numerisches und damit unerlaubtes Zeichen erreicht ist. Nur bis zu einem solchen unerlaubten Zeichen wird die Umformung vorgenommen.

Beispiele:

```
10 A$= "12.89"
20 PRINT VAL(A$)
30 B$= "123A123"
40 PRINT VAL(B$)
50 C$= "56.1.1.1"
60 PRINT VAL(C$)
70 D$="COMPUTER"
80 PRINT VAL(D$)
90 E$="3*4+7"
100 PRINT VAL(E$)
110 F$="-56.1*"
120 PRINT VAL(F$)
RUN
 12.89
 123
 56.1
 0
 3
-56.1
READY.
```

Anmerkung zur Kompatibilität: Die Toleranzgrenze bei der Eingabe ist von Computertyp zu Computertyp verschieden. Bei folgender Eingabe

```
110 f$=".a"
120 PRINT VAL(f$)
```

reagiert zum Beispiel der Commodore-Computer mit dem Ergebnis Ø, während der Schneider CPC einen

TYPE MISMATCH ERROR

meldet, obwohl er ansonsten auch für nicht-numerische Eingaben die Ø als Ergebnis anzeigt.

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  VAL(String)
› COMMODORE VC20,64         :  VAL(String)
› COMMODORE 16,116,PLUS4    :  VAL(String)
› COMMODORE 128             :  VAL(String)
› ENTERPRISE 64,128         :  VAL(String)
› MSX-STANDARD              :  VAL(String)
› SCHNEIDER CPC464          :  VAL(String)
› SCHNEIDER CPC664,CPC6128  :  VAL(String)
› SINCLAIR ZX81             :  VAL String
› SINCLAIR SPECTRUM         :  VAL String
```

VAL$

Befehlsart: Funktion

VAL$ String

Beschreibung: Der Befehl VAL$ entspricht in allen Einzelheiten dem Befehl VAL (siehe dort), bis auf den Unterschied, daß das Ergebnis nicht als Zahl, sondern als String vorliegt.

Beispiel:

```
A$=VAL$"123Peter":PRINT A$
123
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
› SINCLAIR SPECTRUM         :  VAL$ String
```

VARPTR

Befehlsart: Anweisung

VARPTR(**Variablenname**)

Beschreibung: Mit VARPTR läßt sich die Startadresse einer bestimmten Variablen im Speicher feststellen. Dabei wird der dezimale Wert wiedergegeben.

Beispiel:

```
10 A=100
20 S=VARPTR(A)
30 H$=HEX$(S)
40 PRINT H$
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	POINTER(Variable)
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	VARPTR(Variable)
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

VERIFY

Befehlsart: Kommando

VERIFY "**Programmname**",**Zahl**

Beschreibung: Mit Hilfe dieses Kommandos läßt sich überprüfen, ob das Programm **Programmname** korrekt abgespeichert wurde.

Dazu vergleicht man das Programm auf der Kassette bzw. Diskette mit dem Programm im Rechner.

Stellt der Rechner hierbei einen Unterschied fest, gibt er eine Fehlermeldung aus.

Die **Zahl** legt fest, ob man das Programm, welches mit dem Rechnerprogramm verglichen wird, auf Kassette bzw. Diskette abgespeichert hat.

Beispiel:

```
VERIFY "APFELMANN", 8
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
> COMMODORE V64             :  VERIFY "Programmname",Zahl
> COMMODORE 16,116,PLUS4    :  VERIFY "Programmname",Zahl
> COMMODORE 128             :  VERIFY "Programmname",Zahl
> ENTERPRISE 64,128         :  VERIFY #Zahl:"Programmname"
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
> SINCLAIR SPECTRUM         :  VERIFY "Programmname"
```

VOL

Befehlsart: Anweisung

VOL **Zahl**

Beschreibung: Die Lautstärke (engl. "volume") wird für die SOUND- und PLAY-Anweisungen (siehe jeweils dort) durch die **Zahl** festgelegt.

Der Wert der Zahl kann zwischen Ø und 15 liegen. Beim Wert Ø ist dabei die Ton- bzw. Klangausgabe stumm geschaltet; beim Wert 15 wird die größtmögliche Lautstärke erzielt.

Beispiel:

VOL 6

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
> COMMODORE 16,116,PLUS4    :  VOL Zahl
> COMMODORE 128             :  VOL Zahl
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

VPEEK

Befehlsart: Funktion

VPEEK(**Adresse**)

Beschreibung: Mit dem VPEEK-Befehl kann man Daten auslesen, die unter einer vorgeschriebenen Video-RAM-Adresse abgelegt sind. Die **Adresse** muß einen ganzzahligen Wert zwischen Ø und 16383 haben.

Beispiel:

```
10 A=VPEEK(0)
20 PRINT A
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  VPEEK(Adresse)
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

VPOKE

Befehlsart: Funktion

VPOKE **Adresse,Zahl**

Beschreibung: VPOKE schreibt **Zahl**en zwischen Ø und 259 in eine vorgeschriebene **Adresse** des Video-RAMs. Die Adresse muß einen ganzzahligen Wert zwischen Ø und 16383 haben.

Beispiel:

```
10 CLS
20 FOR N=0 TO 100
30 VPOKE N,&FF
40 NEXT
50 FOR F=1 TO 500:NEXT
60 SCREEN 0
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
› MSX-STANDARD	:	VPOKE Adresse,Zahl
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

VPOS

Befehlsart: Funktion

VPOS(#**Kanalnummer**)

Beschreibung: Der Befehl gibt die Zeilenposition des Textcursors wieder. Die **Kanalnummer** muß in jedem Fall angege-

ben werden. Dabei hat der Hauptbildschirm die **Kanalnummer#Ø.**

Beispiel:

```
10 CLS:PRINT "SCHNEIDER BUCH":PRINT VPOS (#0)
20 PRINT "SCHNEIDER BUCH":PRINT VPOS (#0):IF VPOS (#0)<20 THEN GOTO 20
30 END
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :  VPOS(#Kanalnummer)
> SCHNEIDER CPC664,CPC6128  :  VPOS(#Kanalnummer)
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

WAIT

Befehlsart: Kommando

WAIT **Adresse,Byte 2,Byte 3**

Beschreibung: Dieser Befehl läßt das Programm auf den Eingang eines BYTE1 (Zahl zwischen Ø und 255) über die angesprochene **Adresse** warten.

Das empfangene BYTE läßt sich anhand der logischen UND-Funktion (siehe unter AND) mit **Byte 2** verknüpfen. Man erhält als Ergebnis Byte 1 und Byte 2. Soll das empfangene Byte nicht verändert werden, muß **Byte 2** den Wert 255 haben.

Falls man will, läßt sich das empfangene Byte noch über eine exklusive ODER-Funktion (siehe unter XOR) mit **Byte 3** verknüpfen. Dabei wird jedes Bit gemäß folgender Funktionstabelle behandelt:

Bit 1	Ø	1	Ø	1
Bit 3	Ø	Ø	1	1
Ergebnis	Ø	1	1	Ø

Bei Verwendung dieses Befehls ist streng darauf zu achten, daß über die angegebene **Adresse** auch eine Information kommt. Andernfalls kann man aus dem Programm nur noch mit Hilfe eines RESETs herauskommen. Daher sollte man das Programm auf jeden Fall vor Verwendung von WAIT abspeichern.

Beispiel:

WAIT &FCØ7,255,19

Angenommen, man erhält über die angegebene **Adresse** als Byte1 die Zahl 67 (= Ø1ØØØ1Ø1). Dann wird dieses durch **Byte 2**, 255 (= 11111111) mit einem AND verknüpft. Man erhält Byte 1 & Byte 2= 67 (=Ø1ØØØ1Ø1) = Byte 1. Dieses wird nochmals über ein XOR mit Byte 3, 19 (= ØØØ1ØØ11) verknüpft. Byte 1 XOR **Byte 3** ergibt 86 (= Ø1Ø1Ø11Ø).

Anmerkung: Diesen Befehl sollten nur erfahrene Programmierer verwenden.

Anmerkungen zur Kompatibilität: Der WAIT-Befehl hat in den verschiedenen BASIC-Dialekten unterschiedliche Wirkungsweisen und Formate. In einigen Dialekten gibt das Argument hinter dem WAIT-Befehl eine Zeitdauer an, für die das BASIC-Programm wartet.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
> COMMODORE VC20,64	:	WAIT Adresse,Byte2,Byte3
> COMMODORE 16,116,PLUS4	:	WAIT Adresse,Byte2,Byte3
> COMMODORE 128	:	WAIT Adresse,Byte2,Byte3
ENTERPRISE 64,128	:	
> MSX-STANDARD	:	WAIT Adresse,Byte2,Byte3
> SCHNEIDER CPC464	:	WAIT Adresse,Byte2,Byte3
> SCHNEIDER CPC664,CPC6128	:	WAIT Adresse,Byte2,Byte3
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

WAIT DELAY

Befehlsart: Anweisung

WAIT DELAY **Zahl**

Beschreibung: Die Programmausführung wird für eine durch die **Zahl** bestimmte Zeit unterbrochen. Sie gibt die Zeit in Sekunden an, für die der Rechner die Ausführung des Programmes unterbrechen soll. Die **Zahl** kann Werte zwischen 1 und 32767 annehmen.

Beispiel:

```
113Ø PRINT "WILLKOMMEN";
114Ø WAIT DELAY 1Ø
115Ø PRINT "BEI";
116Ø WAIT DELAY 2Ø
117Ø PRINT "MEINEM";
118Ø WAIT DELAY 3Ø
119Ø PRINT "SEHR LANGSAMEN";
12ØØ WAIT DELAY 4Ø
121Ø PRINT "PROGRAMM."
```

Anmerkung zur Kompatibilität: Die Anweisung entspricht dem SLEEP-Befehl (siehe dort).

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	SLEEP Zahl
› ENTERPRISE 64,128	:	WAIT DELAY Zahl
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

WEND

Befehlsart: Anweisung

WEND

Beschreibung: Eine mit dem WHILE-Befehl (siehe dort) erzeugte Schleife wird abgeschlossen.

Beispiel:

```
100 INPUT "eingabe der hoechsten zahl";r
110 a=1
120 WHILE a<r
130 PRINT a,
140 a=a+1
150 WEND
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :  WHILE Bedingung
                               WEND
> SCHNEIDER CPC664,CPC6128  :  WHILE Bedingung
                               WEND
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

WHEN EXEPTION USE

Befehlsart: Anweisung

WHEN EXCEPTION USE **Handlername**
WHEN END

Beschreibung: Diese Anweisung verzweigt ähnlich den Befehlen ON ERROR GOTO, TRAP (siehe dort) in ein Unterprogramm. Diese Verzweigung in ein Unterprogramm wird vorgenommen, wenn ein Fehler auftritt. Mit

WHEN END

wird diese Verzweigung im Falle eines Programmfehlers beendet. Danach wird das BASIC-Programm wieder mit einer Fehlermeldung unterbrochen, falls ein Fehler auftritt.

Die Unterbrechung kann auch durch ein CAUSE EXCEPTION (siehe dort) verursacht werden.

Beispiel:

```
10 REM UNTERBRECHUNG
20 WHEN EXCEPTION USE FEHLER
30   PRINT "GIB EINE ZAHL EIN!"
40   INPUT I
50   IF I>10 THEN CAUSE EXCEPTION 1
60   PRINT "WURZEL AUS"I"GLEICH"SQR(I)
70   IF I<>0 THEN 30
80 END WHEN
90 PRINT "PROGRAMMENDE"
100 END
1000 HANDLER FEHLER
1010   IF EXTYPE=1 THEN PRINT "MIT SO GROSSEN ZAHLEN RECHNE ICH NICHT!":I=1
1020   IF EXTYPE>999 THEN PRINT EXSTRING$(EXTYPE):PRINT "IN ZEILE"EXLINE:I=1
1030   RETRY
1040 END HANDLER
```

In Zeile 2Ø wird die Fehlerverzweigungsroutine eingesetzt. Sie springt in das Unterprogramm HANDLER FEHLER, wenn ein Fehler auftritt. In Zeile 8Ø wird die Routine beendet.

In Zeile 5Ø wird ein "künstlicher" Fehler mit der Nummer 1 erzeugt, wenn die eingegebene Zahl größer als 1Ø ist. Es wird dann in die Routine FEHLER verzweigt.

In Zeile 6Ø kann leicht ein Fehler entstehen, nämlich dann, wenn eine negative Zahl als Argument der Wurzel dienen soll. Auch dann wird in die Routine FEHLER verzweigt.

In Zeile 1ØØØ wird eine Routine definiert, die mit HANDLER FEHLER begonnen und mit END HANDLER in Zeile 1Ø4Ø abgeschlossen wird. Wird die Routine nicht so abgeschlossen, erfolgt eine Fehlermeldung.

In den Zeilen 1Ø1Ø und 1Ø2Ø werden Fehlermeldungen auf dem Bildschirm ausgegeben.

Mit Zeile 1Ø3Ø wird das alte Programm in jener Zeile fortgesetzt, in der es unterbrochen wurde. Diese Zeile wird neuerlich abgearbeitet.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	TRAP Zeilennummer
COMMODORE 128	:	TRAP Zeilennummer
› ENTERPRISE 64,128	:	WHEN EXCEPTION USE Handlername
MSX-STANDARD	:	ON ERROR GOTO Zeilennummer
SCHNEIDER CPC464	:	ON ERROR GOTO Zeilennummer
SCHNEIDER CPC664,CPC6128	:	ON ERROR GOTO Zeilennummer
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

WHILE WEND

Befehlsart: Anweisung

WHILE **logische Bedingung**
WEND

Beschreibung: Mit diesen beiden Befehlen kann man eine Programmschleife aufbauen. Die Schleife wird so lange durchlaufen, bis die **logische Bedingung** des WHILE-Befehls erfüllt ist. Mit dem WEND-Befehl wird die Schleife abgeschlossen.

Anhand dieser beiden Befehle lassen sich Zählschleifen mit verschiedensten Bedingungen aufbauen.

WHILE WEND

Beispiel:

```
10 REM ein verrueckter zaehler
20 CLS
30 i1=48:i2=48:i3=48:i4=48
40 WHILE i4>47
50 WHILE i3<57
60 WHILE i2>47
70 WHILE i1<57
80 PRINT CHR$(30);CHR$(i4);CHR$(i3);CHR$(i2);CHR$(i1)
90 FOR i=0 TO 50:NEXT
100 i1=i1+1
110 WEND
120 i1=48:i2=i2-1
130 WEND
140 i2=57:i3=i3+1
150 WEND
160 i3=48:i4=i4-1
170 WEND
180 i4=57
190 GOTO 40
```

Was ist am Ergebnis dieses Programms "verrückt"? Antwort: Die Einer- und Hunderter-Stellen des Zählers zählen richtig, die anderen verkehrt.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	WHILE Bedingung WEND
› SCHNEIDER CPC664,CPC6128	:	WHILE Bedingung WEND
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

WIDTH

Befehlsart: Anweisung

WIDTH **Zahl**

Beschreibung: Durch diesen Befehl läßt sich die Strichstärke für die grafischen Befehle festlegen (siehe BOX, CIRCLE, DRAW usw.). Wenn die **Zahl** den Wert 1 hat, so erhält man die normale Strichstärke; bei dem Wert 2 entsteht die doppelte.

Beispiel:

```
WIDTH 2
```

Anmerkung: Es gibt noch eine zweite Version des WIDTH-Befehls, die die Druckbreite (Zeilenlänge) festlegt. Also Vorsicht vor Verwechslungen!

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
› COMMODORE 128	:	WIDTH Zahl
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

WIDTH

Befehlsart: Kommando

Beschreibung: Dieser Befehl dient zur Druckeranpassung. Mit WIDTH wird festgelegt, wie viele Zeichen pro Zeile ausgedruckt werden, bevor ein Zeilenvorschub erfolgt. Die **Zahl** gibt die Anzahl der Zeichen wieder.

Beispiel:

```
10 FOR i = 1 TO 20
20 PRINT#8, "x";:WIDTH 5
30 NEXT i
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
> MSX-STANDARD              :  WIDTH Zahl
> SCHNEIDER CPC464          :  WIDTH Zahl
> SCHNEIDER CPC664,CPC6128  :  WIDTH Zahl
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

WINDOW

Befehlsart: Anweisung

WINDOW #**Kanalnummer**,**Zahl 1**,**Zahl 2**,**Zahl 3**,**Zahl 4**

Beschreibung: Mit diesem Befehl kann man bis zu 8 Textfenster auf dem Bildschirm definieren.

Damit sich diese Textfenster ansprechen lassen, tragen sie **Kanalnummern**, deren Wert zwischen Ø und 7 liegen darf. Der Wert Ø bedeutet dabei den Hauptbildschirm.

Ein definiertes Window (window=Fenster) kann man mit den

Befehlen zur Bildschirmausgabe wie PRINT, PAPER, CLS, LOCATE (siehe jeweils dort) über die **Kanalnummer** ansprechen. Wird keine **Kanalnummer** angegeben, so nimmt der Rechner automatisch die **Kanalnummer** #Ø an.

Zahl 1 und **Zahl 2** kennzeichnen die waagrechte Position des Windows. **Zahl 1** gibt die linke Fenstergrenze (="Fensterrahmen") und **Zahl 2** die rechte Fenstergrenze an. Bezugspunkt ist dabei die obere linke Ecke des Bildschirms mit den Koordinaten 1 (Spalte) und 1 (Zeile).

Zahl 2 und **Zahl 3** bezeichnen die senkrechte Position des Windows. **Zahl 3** gibt die oberste Fenstergrenze und **Zahl 4** die unterste Fenstergrenze an.

Beispiel:

WINDOW 1,15,125

läßt den Hauptbildschirm nur noch 5 Zeichen breit sein. Er hat aber immer noch seine volle Höhe. Das Fenster liegt ganz im linken Teil des vorherigen Hauptbildschirm-Ausschnitts. Dabei wird für das Window der neue Bezugspunkt 1,1 die Position **Zahl 1,Zahl 3.**

Beispiel:

```
10 MODE 2
20 WINDOW #1, 41,80,1,25
30 PRINT "test"
40 LIST#1
```

Mit diesem kleinen Programm wird rechts auf dem 8Ø-Zeichen-Bildschirm ein Fenster definiert. In diesem Fenster kann man dann mit LIST 1 das Programm listen und es auf dem Hauptbildschirm starten. So lassen sich einfache Programmfehler finden, da zugleich das Listing und die Ausführung eines Programmes auf dem Bildschirm zu sehen sind.

Anmerkung: Der Befehl eignet sich auch in besonderem Maße für Programme, in denen man mit mehreren Bildschirmen arbeitet, zum Beispiel bei Datenverwaltung.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  WINDOW Zahl1,Zahl2,Zahl3,Zahl4,Zahl5
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :  WINDOW #Kanalnummer,Zahl1,Zahl2,Zahl3,Zahl4
> SCHNEIDER CPC664,CPC6128  :  WINDOW #Kanalnummer,Zahl1,Zahl2,Zahl3,Zahl4
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

WINDOW SWAP

Befehlsart: Anweisung

WINDOW SWAP **Kanalnummer,Kanalnummer**

Beschreibung: Mit diesem Befehl lassen sich zwei durch den WINDOW-Befehl (siehe dort) definierte Fenster vertauschen. So können zum Beispiel alle Ausgaben, die den Hauptbildschirm betreffen, auf ein Fenster ausgegeben werden, und die Ausgaben, die das Fenster betreffen, erscheinen auf dem Hauptbildschirm.

Beispiel:

```
WINDOW SWAP Ø,1
```

WINDOW Ø und WINDOW 1 werden vertauscht.

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :  WINDOW SWAP Zahl1,Zahl2
> SCHNEIDER CPC664,CPC6128  :  WINDOW SWAP Zahl1,Zahl2
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```

WORD$

Befehlsart: Funktion

WORD$(**Zahl**)

Beschreibung: Einem 2-Zeichen-String werden die 2 Bytes zugewiesen, die an der Speicherzelle stehen, die durch die **Zahl** angegeben ist. Das erste Zeichen, das dem String zugewiesen wird, steht an der mit **Zahl** bezeichneten Adresse, das zweite wird durch **Zahl**+1 bezeichnet.

Beispiel:

```
PRINT WORD$(1Ø24)

A$=WORD$(I)
```

Anmerkung: In der Informatik bedeutet der Begriff WORD = Doppelbyte = 2 Bytes.

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
› ENTERPRISE 64,128	:	WORD$(Zahl)
MSX-STANDARD	:	
SCHNEIDER CPC464	:	
SCHNEIDER CPC664,CPC6128	:	
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	

WRITE

Befehlsart: Anweisung

WRITE #**Kanalnummer**,**Ausgabeliste**

Beschreibung: Dieser Befehl wirkt ähnlich dem PRINT-Befehl (siehe dort), jedoch werden bei der Ausgabe die Anführungszeichen der Strings und die Kommata zwischen den einzelnen Variablen mit ausgegeben.

Die **Kanalnummer** benötigen wir nur bei Verwendung von Windows (siehe WINDOW). Mögliche **Kanalnummern** liegen zwischen Ø und 7.

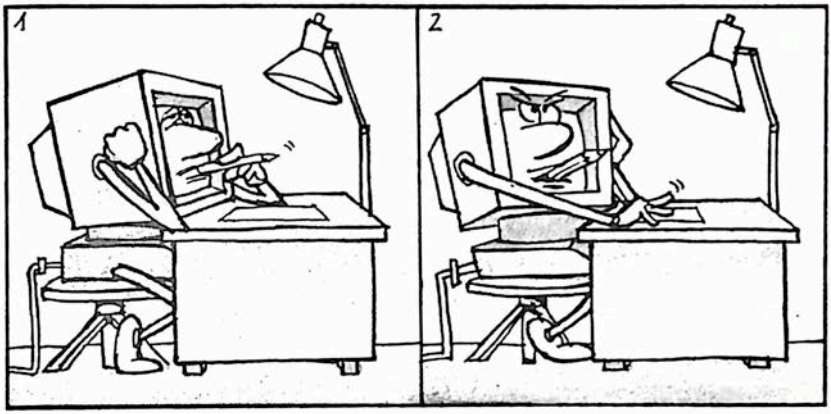
Die **Ausgabeliste** ist die Liste der Variablen, Strings und Zahlen, die ausgegeben werden sollen. Die einzelnen Elemente werden durch Kommata voneinander getrennt.

Beispiel:

```
WRITE "schneiderbuch",a,5
"schneiderbuch",Ø,5
READY
```

Kompatibilitätsliste:

ATARI 600,800,130XE	:	
COMMODORE VC20,64	:	
COMMODORE 16,116,PLUS4	:	
COMMODORE 128	:	
ENTERPRISE 64,128	:	
MSX-STANDARD	:	
› SCHNEIDER CPC464	:	WRITE #Zahl,Ausgabeliste
› SCHNEIDER CPC664,CPC6128	:	WRITE #Zahl,Ausgabeliste
SINCLAIR ZX81	:	
SINCLAIR SPECTRUM	:	



XIO

Befehlsart: Anweisung

XIO **Zahl 1,Zahl 2,Zahl 3,Zahl 4,"Textausdruck"**

Beschreibung: Mit dem XIO-Befehl kann man mehrere Ein- und Ausgabefunktionen durchführen.

Zahl 1 gibt mit ihrem Wert zwischen Ø und 255 an, welche Operation durchzuführen ist.

Die nachstehende Tabelle zeigt einen Überblick zu den Möglichkeiten, die zumeist durch einen BASIC-Befehl ersetzt werden müssen.

Zahl 1	Operation	**Zahl 1**	Operation
3	OPEN	5	INPUT
7	GET	9	PRINT
11	PUT	12	CLOSE
13	STATUS	17	DRAWTO
18	Ausfüllen einer Fläche	32	Umbenennen einer Disk-Datei
33	Löschen einer Disk-Datei	35	Disk-Datei-Schreibschutz
36	Schreibschutz entfernen	37	POINT
38	NOTE	255	Disk formatieren

Die **Zahl 2** gibt die Nummer des Kanals an, mit dem operiert werden soll. Die Zahl kann zwischen 1 und 7 liegen. In der Regel muß man diesen Kanal mit Hilfe des OPEN-Befehls (siehe dort) öffnen. Ausnahme ist die **Zahl 1**=3, da dieses Format den BASIC-Befehl OPEN simuliert.

Zahl 3 und **Zahl 4** geben zusätzliche Parameter für die Operation an. Sie werden in der Mehrzahl der Fälle nicht benötigt und lassen sich als Ø ansetzen.

Der **Textausdruck** ist eine Zeichenkette, die zusätzliche Informationen, wie zum Beispiel den Disk-Dateinamen, enthält.

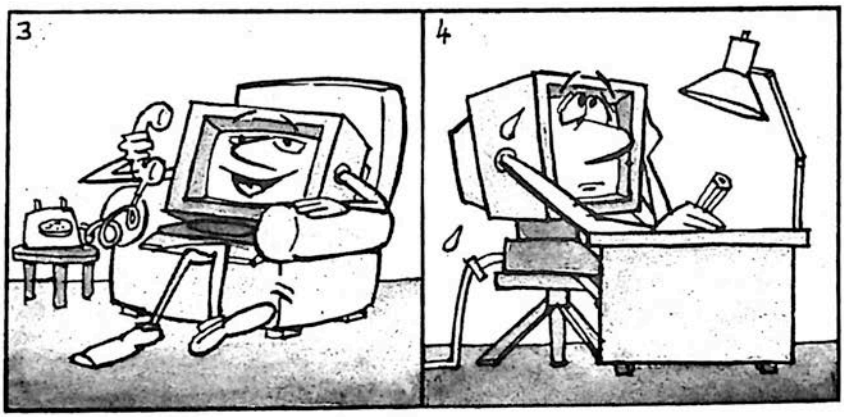
Beispiel:

```
XIO 33, 1,Ø,Ø,"D:VERSUCH"
```

bewirkt das Löschen der Disk-Datei VERSUCH.

Kompatibilitätsliste:

```
› ATARI 600,800,130XE       :  XIO Zahl1,#Zahl2,Zahl3,Zahl4,String
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
  COMMODORE 128             :
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
  SCHNEIDER CPC464          :
  SCHNEIDER CPC664,CPC6128  :
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```



XOR

Befehlsart: Funktion

Zahl 1 XOR Zahl 2

Beschreibung: Die logische Funktion XOR, das sogenannte EXKLUSIV ODER, läßt beim Vergleich zweier Argumente in Bits das Ergebnis 1 bzw. "wahr" sein, wenn eines der beiden Argumente 1 bzw. "wahr" ist.

Beispiel:

```
PRINT 14 XOR 15
 1
READY
```

XOR vergleicht die beiden Argumente, um zu einem Ergebnis zu kommen:

```
  PRINT 15 XOR 14
```

ergibt

```
  1
```

weil

```
                          15 = 0000000000001111
             XOR
                          14 = 0000000000001110

Ergebnis:                  1 = 0000000000000001
```

XOR

In eine logische Operation übersetzt bedeutet dieser Vergleich:

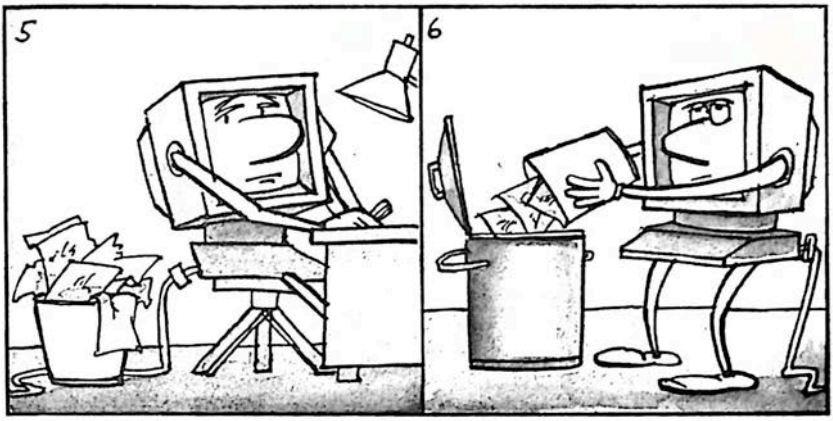
	XOR		ergibt
nicht wahr (Ø)		nicht wahr (Ø)	nicht wahr (Ø)
nicht wahr (Ø)		nicht wahr (Ø)	nicht wahr (Ø)
nicht wahr (Ø)		nicht wahr (Ø)	nicht wahr (Ø)
nicht wahr (Ø)		nicht wahr (Ø)	nicht wahr (Ø)
nicht wahr (Ø)		nicht wahr (Ø)	nicht wahr (Ø)
nicht wahr (Ø)		nicht wahr (Ø)	nicht wahr (Ø)
nicht wahr (Ø)		nicht wahr (Ø)	nicht wahr (Ø)
nicht wahr (Ø)		nicht wahr (Ø)	nicht wahr (Ø)
nicht wahr (Ø)		nicht wahr (Ø)	nicht wahr (Ø)
nicht wahr (Ø)		nicht wahr (Ø)	nicht wahr (Ø)
nicht wahr (Ø)		nicht wahr (Ø)	nicht wahr (Ø)
nicht wahr (Ø)		nicht wahr (Ø)	nicht wahr (Ø)
wahr (1)		wahr (1)	nicht wahr (Ø)
wahr (1)		wahr (1)	nicht wahr (Ø)
wahr (1)		wahr (1)	nicht wahr (Ø)
wahr (1)		nicht wahr (Ø)	wahr (1)

Beispiel:

```
10 INPUT x
20 INPUT y
30 z=x XOR y
40 PRINT BIN$(x,8)
50 PRINT BIN$(y,8)
60 PRINT "-----------"
70 PRINT BIN$(z,8)
80 GOTO 10
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :
> COMMODORE 128             :  Zahl1 XOR Zahl2
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :  Zahl1 XOR Zahl2
> SCHNEIDER CPC664,CPC6128  :  Zahl1 XOR Zahl2
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```



XPOS

Befehlsart: Funktion

XPOS

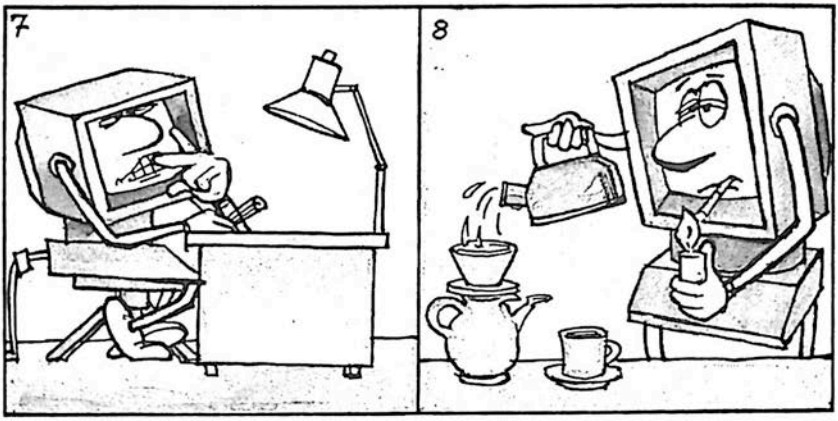
Beschreibung: Die Funktion XPOS gibt die augenblickliche Stellung des Grafikcursors in x-Richtung (Spalten) zurück.

Beispiel:

```
PRINT XPOS
 Ø
READY
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE          :
  COMMODORE VC20,64            :
  COMMODORE 16,116,PLUS4       :  RDOT(0)
  COMMODORE 128                :  RDOT(0)
  ENTERPRISE 64,128            :
  MSX-STANDARD                 :
› SCHNEIDER CPC464             :  XPOS
› SCHNEIDER CPC664,CPC6128     :  XPOS
  SINCLAIR ZX81                :
  SINCLAIR SPECTRUM            :
```



YPOS

Befehlsart: Funktion

YPOS

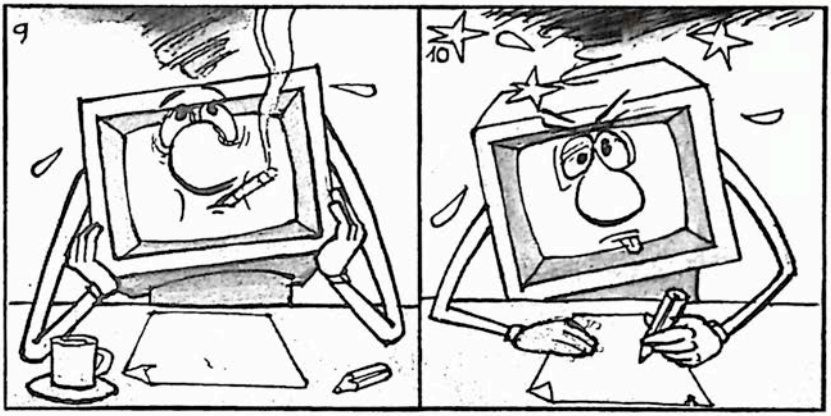
Beschreibung: Die Funktion YPOS gibt die augenblickliche Stellung des Graficursors in y-Richtung (Zeilen) zurück.

Beispiel:

```
10 CLS
20 DRAW 100,90:PRINT XPOS;YPOS
30 DRAW 100,250:PRINT XPOS;YPOS
40 DRAW 250,250:PRINT XPOS;YPOS
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE       :
  COMMODORE VC20,64         :
  COMMODORE 16,116,PLUS4    :  RDOT(1)
  COMMODORE 128             :  RDOT(1)
  ENTERPRISE 64,128         :
  MSX-STANDARD              :
> SCHNEIDER CPC464          :  YPOS
> SCHNEIDER CPC664,CPC6128  :  YPOS
  SINCLAIR ZX81             :
  SINCLAIR SPECTRUM         :
```



ZONE

Befehlsart: Kommando

ZONE **Zahl**

Beschreibung: Mit diesem Befehl wird die Breite jener Druckzonen geändert, die die PRINT-Anweisung (siehe dort) verwendet. Der Rechner benutzt als Standardwert 13. Der Wert der **Zahl** kann zwischen 3 und 255 liegen. Dabei ist allerdings folgendes zu beachten:

Sollen etwa bei

PRINT **Zahl** 1,**Zahl** 2,**Zahl** 3

die Abstände bei der Ausgabe korrekt und überall gleich sein, so muß der Wert der **Zahl** im Format des ZONE-Befehls um mindestens 3 größer sein als die Stellen-Anzahl der größten Zahl. Andernfalls kommt es zu falschen Abständen bei der Ausgabe.

Dabei bezieht sich der Abstand auf die jeweils ersten Zeichen der auszugebenden Zeichenketten.

Bei der Ausgabe von Strings muß die **Zahl** im Format des ZONE-Befehls nur um mindestens 2 größer sein als die Anzahl der Zeichen im größtmöglichen String.

ZONE

Beispiel:

```
ZONE 8
PRINT 12345,123,12
```

Kompatibilitätsliste:

```
  ATARI 600,800,130XE          :
  COMMODORE VC20,64            :
  COMMODORE 16,116,PLUS4       :
  COMMODORE 128                :
  ENTERPRISE 64,128            :
  MSX-STANDARD                 :
> SCHNEIDER CPC464             :  ZONE Zahl
> SCHNEIDER CPC664,CPC6128     :  ZONE Zahl
  SINCLAIR ZX81                :
  SINCLAIR SPECTRUM            :
```

ON INTERVAL GOSUB
ON KEY GOSUB
ON SPRITE GOSUB
ON SQ GOSUB
ON STOP GOSUB
ON STRIG GOSUB
OPEN
OPENIN
OPENOUT
OPTION ANGLE
OR
ORD
ORIGIN
OUT
PAD
PADDLE
PAINT
PAPER
PDL
PEEK
PEN
PEN
PI
PING
PLAY
PLOT
PLOTR
POINT
POINTER
POKE
POP
POS
POSITION
POT
PRESET
PRINT
PRINT#
PRINT USING
PROGRAM
PSET
PTRIG
PUDEF
PUT
PUT SPRITE
RAD
RAD
RANDOMIZE
RCLR
RDOT
READ
RECORD
REDIRECT
RELEASE
REM
REM
REMAIN
RENAME
RENUM
RENUMBER
RESTORE
RESUME
RETRY
RETURN
RGR
RIGHT$
RLUM
RND
ROUND
RREG
RSPCOLOR
RSPPOS
RSPRITE
RTRIM$
RUN
RUN
RWINDOW
SAVE
SCALE
SCNCLR
SCRATCH
SCREEN
SCREEN$
SCROLL
SEC
SELECT
SET
SETCOLOR
SGN
SIN
SINH
SIZE
SLEEP
SLOW
SOUND
SPACE$
SPC
SPEED INK
SPEED KEY
SPEED WRITE
SPEEK
SPOKE
SPRCOLOR
SPRDEF
SPRITE
SPRITE$
SPRITE ON/OFF
SPRSAV
SQ
SQR
SSHAPE
START
STASH
STATUS
STEP
STICK
STICK
STOP
STOP ON/OFF
STR$
STRIG
STRIG ON/OFF
STRING
STRING$
SWAP
SWAP
SYMBOL
SYMBOL AFTER
SYS
TAB
TAG
TAGOFF
TAN
TANH
TEMPO
TEST
TESTR
TEXT
THEN
TIME
TIME
TIME$
TO
TOGGLE
TRACE ON/OFF
TRAP
TROFF
TRON
TRUNCATE
TYPE
UBOUND
UCASE$
UNT
UNTIL
UPPER$
USING
USR
VAL
VAL$
VARPTR
VERIFY
VOL
VPEEK
VPOKE
VPOS
WAIT
WAIT DELAY
WEND
WHEN EXCEPTION USE
WHILE WEND
WIDTH
WIDTH
WINDOW
WINDOW SWAP
WORD$
WRITE
XIO
XOR
XPOS
YPOS
ZONE

Handbuch der BASIC-Dialekte

Jeder Heimcomputer muß in seiner eigenen Sprache angesprochen werden – nämlich in seinem speziellen BASIC-Dialekt. Dieses Buch bietet über 400 alphabetisch geordnete Stichworte (mit Anwendungsbeispielen) zu den BASIC-Versionen der am weitesten verbreiteten Fabrikate.

Es zeigt, welche BASIC-Befehle der jeweilige Computertyp versteht.

Dieses Vergleichslexikon bietet wichtige Hilfen für den Anfänger, für den Computer-Umsteiger sowie für jeden, der Programme tauschen möchte; vor allem jedoch für jeden, der die Bandbreite seiner BASIC-Version voll ausnützen will.

Mit Hilfe dieses Handbuchs kann man:

- ein Programm in mehreren Versionen schreiben,
- Programme auf ihre Übertragbarkeit abklopfen,
- besonders exotische BASIC-Befehle einer bestimmten Version durch Hilfsprogramme auf einem anderen Modell ersetzen.

Den Überblick erleichtert außerdem ein nach Computertypen aufgeschlüsseltes Befehlsregister.

Folgende Homecomputer werden geschildert:

Atari 130 XE	Commodore 64	Schneider CPC 464
Atari 600	Commodore 116	Schneider CPC 664
Atari 800	Commodore 128	Sinclair ZK 81
Commodore + 4	Enterprise PHC 64/128	Sinclair Spectrum
Commodore 16	Philips MSX	Sinclair QL
Commodore VC 20	Sanyo MSK	Sony MSX

ISB N 3-505 09206-1

SCHNEIDER VERLAG